THE DERIVE - NEWSLETTER #66

ISSN 1990-7079

THE BULLETIN OF THE DERIVE USER GROUP + CAS-TI

C o n t e n t s:

June 2007

GREAT NEWS: Calculus Made Easy version 9.0 is now available.

Calculus Made Easy v9.0 has become the ultimate educational tool to study Calculus.

View the slide shows at www.ti89.com to view step by step solutions in all areas of Calculus.

What is new in v9.0?

1) Step by Step Limits
2) Step by Step Differential Equations
3) Step by Step Implicit Differentiation
4) Step by Step Sequences and Series / Convergence Intervals
5) Step by Step Area Approximation
6) Step by Step Partial Fractions and Trig Substitution
7) Step by Step Synthetic Division and Composition of Functions & much more

No other software covers Calculus I (Calc AB) , II (Calc BC) , III (multivariable), Differential Equations, Vector Calculus and Differential Geometry!!

Jetzt auch auf deutsch: **http://www.ti89.com/store_eur.htm**

Product information at: **www.ti89.com/documentation.htm** .
To Upgrade: Simply click on <Upgrade> at **www.ti89.com/store.htm** .
Contact: support@ti89.com

At the CAME5-conference which was held in Pécs, Hungary, I met Prof. András Ringler from the University of Szeged. He found a beautiful geometric solution of quadratic equations. Your are invited to follow his ideas.

Goto: **www.mozaik.info.hu/homepage/mozaportal/Matematika.php**

To activate the program click on the icon **Gondolkozzunk "görögül"**, then **Megnyitás**, then **Ringler Andras G** …, and finally **GEOMexe**.

If you are looking for an exciting application of matrix calculation in connection with eigenvectors which is attractive for students, then download

"An Illustrated Guide to the ANALYTIC HIERARCHY PROCESS"

presented by Dr. Rainer Haas and Dr. Oliver Meixner from the University of Natural Resources and Applied Life Sciences, Vienna.

http://www.boku.ac.at/mi/ahptutorial.pdf

Liebe DUG-Mitglieder,

Mitten im Sommer kann ich Ihnen den DNL#66 vorlegen. Ich wollte eigentlich mehr Beiträge in diesen DNL aufnehmen, aber, wie Sie selbst sehen können, machen drei Artikel gemeinsam mit dem Userforum einen vollen DNL aus.

Hubert Weller stellt wieder einmal die Bézier Kurven in den Mittelpunkt seines Aufsatzes. Diese Kurven sind schon öfters im DNL behandelt worden. Dennoch sind sie immer wieder interessant, stellen sie doch eine so schöne Verbindung zwischen Vektorrechung, Parameterdarstellung von Kurven, dynamischer Geometrie und modernem Design in der Ebene und im Raum dar.

Wolfgang Pröpper zeigt in seinem Beitrag zur Integration einerseits die Möglichkeiten des TI-NSpire zur Verbindung verschiedener Darstellungsformen von mathematischen Inhalten und andererseits eine interessante Möglichkeit, Treppenfunktionen zur Definition von Riemannsummen zu erstellen.

Ich lade bei dieser Gelegenheit alle ein, die sich bereits mit dem TI-NSpire beschäftigt haben, über ihre Erfahrungen zu berichten.

Vom schönsten Ende der Welt, von Neuseeland, hat mir G P Speck einen Artikel geschickt, der eine feine Fortsetzung zu den Sudoku-Beiträgen von Johann Wiesenbauer bildet. Die Challenger-Probleme sind eine reizvolle Art von Denksportaufgaben. G P Speck löst sie mit Derive und verwendet eine interessante Verknüpfung von Textanweisungen und auszuführenden DERIVE-Kommandos.

Die letzte Seite gilt den von mir im letzten DNL angesprochenen Flächen, die in einer renommierten deutschen Zeitung auf einer Doppelseite aufgelistet und abgebildet worden sind. Hoffentlich finden Sie die Gegenüberstellung der verschiedenen Darstellungen auch so reizvoll wie ich.

Wenn sie aufmerksam die Liste der unveröffentlichten Beiträge lesen, werden sie merken, dass drei Titel entfernt wurden – der Inhalt dieses DNL – aber schon wieder vier neue hinzugekommen sind. Die Aktivität der DUG-Mitglieder ist wirklich erfreulich. Herzlichen Dank dafür.

Abschließend möchte ich Sie auf die Info-Seite hinweisen und verbleibe mit den besten Wünschen für einen schönen Sommer

Ihr

Dear DUG Members,

It is midsummer and I have the pleasure to offer DNL#66. I intended to include more contributions but as you can see, three extended articles together with the User Forum make the DNL full.

Hubert Weller treats once more the Bézier Curves in his paper. These curves appeared earlier in the DNL. However, they are always interesting forming a fine connection between calculation with vectors, parameter representation of curves, dynamic geometry and modern design in plane and space.

Wolfgang Pröpper shows in his article on one hand the possibilities of linking various representation forms of mathematical contents with TI-NSpire and on the other hand an interesting way for creating stepfunctions to define Riemann sums.

At this occasion I´d like to invite all of you who have worked with the TI-NSpire to report about their experiences.

G P Speck sent an article from the "most beautiful end of the world", New Zealand, which is a fine sequel to Johann Wiesenbauer´s findings on Sudokus. The Challenger problems are nice brain-twisters. G P Speck solves them supported by Derive and uses an interesting combination of instructions to read and Derive commands to execute within the Derive file.

The last page is dedicated to the surfaces which I mentioned in the last DNL. They were listed and shown in a renowned german newspaper. I hope that you will find the comparison of the various representations as inspiring as I do.

Reading carfully the list of unpublished contributions you will note, that three topics have been removed – contents of this DNL – but four new ones have been included. The activity of the DUG-members is really impressive. Many thanks for that.

Finally I´d like to draw your attention to the information page und remain with best wishes for an enjoyable summer

Sincerely yours

Download all *DNL-DERIVE*- and TI-files from

http://www.austromath.at/dug/

http://www.bk-teachware.com/main.asp?session=375059

The *DERIVE-NEWSLETTER* is the Bulletin of the *DERIVE* & CAS-*TI User Group*. It is published at least four times a year with a contents of 40 pages minimum. The goals of the *DNL* are to enable the exchange of experiences made with *DERIVE*, *TI*-CAS and other CAS as well to create a group to discuss the possibilities of new methodical and didactical manners in teaching mathematics.

Editor: Mag. Josef Böhm
D´Lust 1, A-3042 Würmla
Austria
Phone/FAX: 43-(0)2275/8207
e-mail: nojo.boehm@pgv.at

Contributions:

Please send all contributions to the Editor. Non-English speakers are encouraged to write their contributions in English to reinforce the international touch of the *DNL*. It must be said, though, that non-English articles will be warmly welcomed nonetheless. Your contributions will be edited but not assessed. By submitting articles the author gives his consent for reprinting it in the *DNL*. The more contributions you will send, the more lively and richer in contents the *DERIVE* & CAS-*TI Newsletter* will be.

Next issue: September 2007
Deadline 15 August 2007

Preview: Contributions waiting to be published

Some simulations of Random Experiments, J. Böhm, AUT, Lorenz Kopp, GER
Wonderful World of Pedal Curves, J. Böhm
Another Task for End Examination, J. Lechner, AUT
Tools for 3D-Problems, P. Lüke-Rosendahl, GER
ANOVA with *DERIVE* & *TI*, M. R. Phillips, USA
Financial Mathematics 4, M. R. Phillips
Hill-Encription, J. Böhm
Farey Sequences on the *TI,* M. Lesmes-Acosta, COL
Simulating a Graphing Calculator in *DERIVE*, J. Böhm
Henon & Co, J. Böhm
Are all Bodies falling equally fast, J. Lechner
Do you know this? Cabri & CAS on PC and Handheld, W. Wegscheider, AUT
An Interesting Problem with a Triangle, Steiner Point, P. Lüke-Rosendahl, GER
Overcoming Branch & Bound by Simulation, J. Böhm, AUT
Diophantine Polynomials, D. E. McDougall, Canada
Graphics World, Currency Change, P. Charland, CAN
Precise Recurring Decimal Notation, P. Schofield, UK
Problems solved using the TI-Nspire, K. Stulens, BEL
Cubics, Quartics – interesting features, T. Koller & J. Böhm
Logos of Companies as an Inspiration for Math Teaching
Exciting Surfaces in the FAZ
Centroid of a Triangle – Recursively, D. Sjöstrand, SWE
BooleanPlots.mth, P. Schofield, UK
What is hiding in Dr. Pest? B. Grabinger, GER
Truth Tables on the TI, M. R. Phillips
Advanced Regression Routines for the TIs, M. R. Phillips
Directing Our Suspicions with AHP, C. Leinbach, USA
Embroidery Patters, H. Ludwig, GER

and Setif, FRA; Vermeylen, BEL; Leinbach, USA; Baumann, GER; Keunecke, GER, and others

Impressum:
Medieninhaber: *DERIVE* User Group, A-3042 Würmla, D´Lust 1, AUSTRIA
Richtung: Fachzeitschrift
Herausgeber: Mag.Josef Böhm

A Useful Tipp for Calculatig Sums and Products

Dear Hannes,
Do you know why expression #12 does not return the requested product (expression #11 does)?
Regards,
Josef

```
        prim(n) :=
          If NEXT_PRIME(n - 1) = n
#10:         n
             1

        100
#11:     ∏   prim(k) = 2305567963945518424753102147331756070
        k=2

        1000
#12:     ∏    prim(k) = 1
        k=2
```

<u>Johann Wiesenbauer, Vienna</u>

Dear Josef,
yes, I know this phenomenon. Calculating sums and products of self defined functions with more than 100 addends or factors it happens very often that DERIVE has problems simplifying the sum or product. (Try 102 and then 103 as upper bound.) But there is an easy trick to overcome this deficiency: include the increment which is 1 in your case as an additional 4th parameter.
Regards,
Hannes

```
∏(prim(k), k, 2, 1000, 1) =

  19590340644999083431262508198206381046123972390589368223882605328968666316379870~
  66185195164878948232159622955911543601914918952972521526672829228299085264902336~
  27313924040179391420109582613936349594714837571967216722434100671185162276611331~
  35192488848989914892157188308679896875137439519338903968094905549750386407106033~
  83658666068353920101163591790003990449506520329974954298599313466981480531847408~
  0581207891125910
```

Power of a Matrix

<u>Michael Saake</u>

Hello Derivers,

With DERIVE 6 I get definitely a wrong result calculating Matrix Power A^n.
For example: if A : =[0.5 , 0.5 ; –0.5 , 0.5] then DERIVE simplifies A^n to

$$A^n = \begin{bmatrix} 2^{-n} & 2^{-n} \\ 2^{-n}\cdot(-1)^n & 2^{-n} \end{bmatrix}$$

which is wrong for n > 1. With DERIVE 5 A^n stays A^n. (This problem has been mentioned earlier, but was not solved - as I remember)

The problem is not to find a different way to get the right result for each power A^n or a specific integer n (VECTOR or ITERATE may do), but my students might believe DERIVE´s result and then come to a false conclusion about i.e. lim(A^n). That´s why I think this should not happen and Vers. 5 handles it correct.

Any suggestions?
Thanks
Michael Saake

Johann Wiesenbauer, Vienna

Hi Michael,

No offence meant, but isn't it a little "blue-eyed" to assume that Derive could give a general expression of A^n for a any matrix A? At least, that's what I would say if one my students tried to input such an expression. What Derive really does you could see in the following example

$$\#3:\quad U := \begin{bmatrix} 2 & 3 & 5 \\ 7 & 11 & 13 \\ 17 & 19 & 23 \end{bmatrix}$$

$$\#4:\quad U^n = \begin{bmatrix} 2^n & 3^n & 5^n \\ 7^n & 11^n & 13^n \\ 17^n & 19^n & 23^n \end{bmatrix}$$

In short, it computes the matrix of n-th powers, which is usually not the n-th power of the matrix U. Even though this feature might be occasionally useful, hence I prefer it to letting U^n unsimplified.

Cheers,
Johann

Michael Saake

Thanks Johann,

I think, I understand now not to see this as a bug, but as an intentional different definition of "Matrix-Powers". I could deal with it ...

But to get some of this "blue-eyed" students, that we might have to deal with, in a useful discussion using their own mind to argue about the limits of CAS-Computing, I found it so long very helpful when DERIVE does sometimes nothing (i.e. A^n remains A^n) instead of DERIVE does something which is not compatible with that, what I assume DERIVE will do.

So long
Michael

Johann Wiesenbauer

Hello Michael,

There is one more argument that might convince you that the current definition of A^n for a matrix A, where n is a variable and not a number, is quite useful sometimes. For this you have to replace the "blue-eyed" student in my previous example by a very clever one. He knows that by using a matrix T, whose columns are eigenvectors of A, it might be possible to get a diagonal matrix D, defined by $D= T^{-1} A\ T$. Now for D this "Derive-exponentiation" with a general n is valid. Hence, $A^n = T\ D^n\ T^{-1}$.

If the student is good at programming in Derive as well, he might even write a small program that does all those steps automatically. What do you think of it?

Cheers,
Johann

Michael Saake

Hello Johann,
I´ve just applied it to my earlier example A=[0.5,-0.5 ; 0.5,0.5] and got what I like to get:

$A^n = T \cdot D^n \cdot T^{-1}$ = [?, ?; ?, ?]

But we use DERIVE at high-school-level as an educational tool or calculus servant.

Thanks
Michael

Johann Wiesenbauer

Hello Michael,
Ok, in the meantime I had a go at it myself, and you are right, things are slightly more complicated when writing a program. In particular, you have to specify n, as can be seen in the program below:

```
       mypower(a, t_) :=
         Prog
           t_ := APPEND(VECTOR(EXACT_EIGENVECTOR(a, e_)', e_, EIGENVALUES(a)))'
#1:        n :ε Integer [0, ∞)
           a := t_^(-1)·a·t_
           t_·a^n·t_^(-1)
```

#2: $$\text{mypower}\begin{bmatrix} 0.5 & 0.5 \\ -0.5 & 0.5 \end{bmatrix} = \begin{bmatrix} 2^{-n/2}\cdot \text{COS}\left(\dfrac{\pi\cdot n}{4}\right) & 2^{-n/2}\cdot \text{SIN}\left(\dfrac{\pi\cdot n}{4}\right) \\ -2^{-n/2}\cdot \text{SIN}\left(\dfrac{\pi\cdot n}{4}\right) & 2^{-n/2}\cdot \text{COS}\left(\dfrac{\pi\cdot n}{4}\right) \end{bmatrix}$$

Note that I wrote this program only for the matrix at issue, i.e. for didactical purposes, as the existence of a basis consisting of orthogonal eigenvectors is not always garanteed (it is though, if A commutes with A`, as in your case) and even if it exists, it is very hard or impossible for bigger matrices to compute the exact eigenvalues. This may also be the main reason, why this routine is not implemented in Derive.
Cheers,
Johann

Michael Saake

Thanks, Johann,
I think you made it perfectly clear why to handle matrix powers carefully using CAS. I´ll further point it out to students exploring maths with DERIVE.
Michael

Danny Ross Lunsford

There should definitely be an intrinsic function to compute A^n and exp(A) for general matrices. These things are absolutely fundamental. The TI92+ can exponentiate matrices and I wish Derive had that capability. I use Derive for quick and dirty calculations with the Clifford algebras that appear in field theory - similarity transformations come up all the time.

-drl

to be continued on page 44

Mathematik und Design? – de Casteljau-Algorithmus, Bézier-Kurven und Flächen im Raum

Hubert Weller, Lahnau, hubert.weller@schule.uni-giessen.de

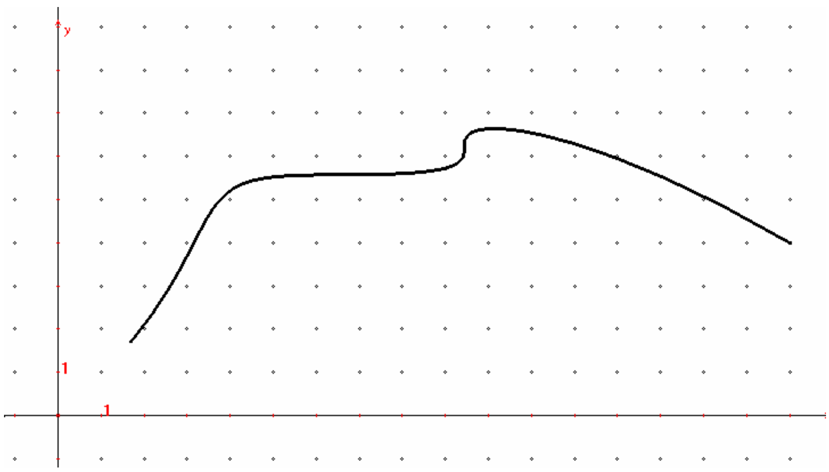
1. Mathematik und Design – Problemstellung und historische Anmerkungen

Die heutige Entwicklung in vielen Teilen der Wirtschaft ist ohne den (massiven) Einsatz von Computern nicht mehr denkbar. Diese Entwicklung hat erst um 1960 begonnen. Dabei hat auf der einen Seite die Mathematik den Einsatz der Computer erst ermöglicht, auf der anderen Seite hat aber der Wunsch, den Computer bei der Schaffung und Herstellung neuer Formen zu nutzen, die Entwicklung völlig neuer mathematischer Methoden erforderlich gemacht. Um mathematische Methoden im Zusammenhang mit der Beschreibung geometrischer Formen geht es in diesem Beitrag.
Ein Designer hat Visionen! Er entwirft eine Form z.B. indem er auf dem Papier einen Entwurf anfertigt, der dann realisiert werden soll. Die Techniker und Ingenieure stehen vor dem Problem, die Form so herzustellen, dass sie jederzeit reproduziert und beliebig oft kopiert werden kann. Noch bis 1960 wurde ein so genanntes „master model" hergestellt und benutzt. Dies hatte aber viele Nachteile:

- die Vorlage nutzt sich ab,
- sie ist schlecht zu transportieren,
- die Reproduktion ist ungenau, ...

Zusätzlich war durch die Fortschritte im CNC-Bereich der Wunsch entstanden, diese Formen mit mathematischen Mitteln zu beschreiben, damit durch die numerischen Daten die computergesteuerten Maschinen bei der Produktion eingesetzt werden konnten. Ganz vereinfacht lässt sich das Problem so beschreiben:

> ***Ein Designer entwirft eine Kurve, die wir mathematisch beschreiben sollen, so dass sie im PC bearbeitet werden kann.***



Dass dies mit den uns bekannten Methoden nicht so einfach ist, kann sich jeder sofort vorstellen!

The artist designs a form, which should be described mathematically in order to work with it on the PC and later on in a computer controlled production process. One can imagine that this is not so easy to do applying the methods which are wellknown to the students.

Besser wäre es, wenn wir dem Designer ein Werkzeug zur Verfügung stellen können, das es ihm erlaubt, solche Kurven am Computer zu entwerfen (ohne die Mathematik verstehen zu müssen) und deren Verlauf wir durch wenige Daten reproduzieren können.

Aufgabenstellung:

Konstruiere ein Werkzeug, mit dem am Computer schöne Kurven erzeugt werden können, die man dann durch wenige Daten algebraisch (und damit auch numerisch) beschreiben kann.

Problem:

Develop a tool for creating beautiful curves on the PC which can be defined algebraically (and subsequently numerically) by only a few data.

An der Lösung dieses Problems wurde um 1960 sowohl in der amerikanischen Flugzeugindustrie als auch unabhängig voneinander in der französischen Automobilindustrie bei Citroen und Renault gearbeitet.

P.de Casteljau (1959) bei Citroen und *P.Bézier* (1961) bei Renault entwickelten unter strenger Geheimhaltung durch die Firmen die notwendige Theorie. Da die Arbeiten von de Casteljau nicht veröffentlicht wurden, werden die Kurven heute als Bézierkurven bezeichnet. Der geometrische Konstruktionsalgorithmus ist nach de Casteljau benannt.

Heutzutage haben wir die Möglichkeit, mit Dynamischer Geometrie Software (CABRI) den de-Casteljau-Algorithmus nachzuvollziehen und damit das Werkzeug für den Designer selbst herzustellen. Darüber hinaus haben wir mit einem Computer-Algebra-System (DERIVE) ein Werkzeug zur Beschreibung, Darstellung und Erzeugung der Kurven und Flächen im Raum.

P.de Casteljeau (Citroen, 1959) and P. Bézier (Renault, 1961) developed the theory. Nowadays we can reproduce the algorithm using Dynamic Geometry (CABRI) and describe the resulting curves using any CAS (DERIVE).

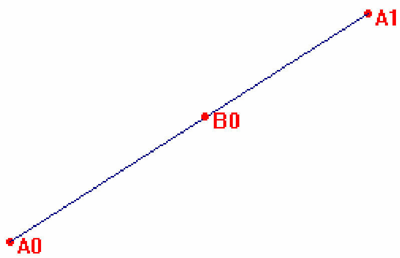
2. De Casteljaus Algorithmus – Der geometrische Zugang

(*We show the procedure using the excellent DGS-program CABRI. In the appendix you can find some screen shots using the pre-release version of GeoGebra. Josef*)

Wir konstruieren zunächst eine Strecke A0A1 und einen Punkt B0 auf der Strecke.

Dieser lässt sich im Zugmodus auf der Strecke verschieben.

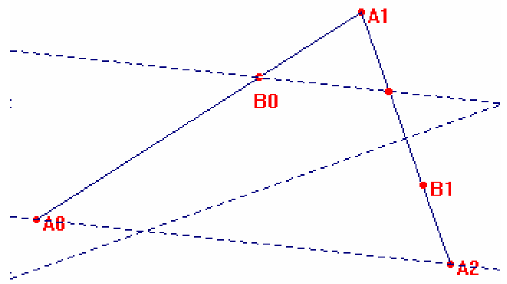
We define segment A0A1 with point B0 on it. B0 can be moved on the segment.



Im Punkt A1 wird eine weitere Strecke A1A2 gezeichnet. Auf dieser Strecke soll ein Punkt B1 konstruiert werden, der die Strecke A1A2 im selben Verhältnis teilt wie der Punkt B0 die Strecke A0A1.

Dazu erinnern wir uns an den Strahlensatz und beachten, dass noch an der Mittelsenkrechten (am Mittelpunkt von A1A2) gespiegelt werden muss, damit B1 in der Nähe von A2 ist, wenn B0 in der Nähe von A1 ist.

Wir „verstecken" alle bei der Konstruktion benutzten Hilfslinien und –punkte.



We add segment A1A2 and transfer the ratio A0A1/A0B0 (incuding a reflection wrt the perpendiular bisector) in order to obtain point B1. Then we hide the auxiliary objects.

Im Zugmodus kann B0 auf der Strecke verschoben werden – B1 bewegt sich entsprechend mit.

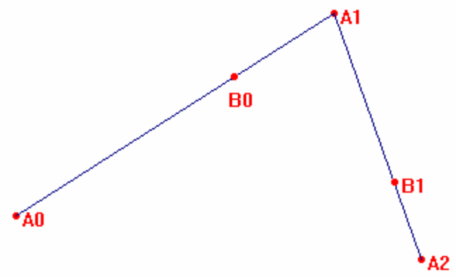
Auch A0, A1 und A2 sind frei gewählte Punkte – sie können mit der Maus beliebig verändert werden.

Die ganze Konstruktion definieren wir als ein **Makro**:

Startobjekte sind A0 , A1 , B0 und A2.

Zielobjekt ist B1.

Als Makronamen benutzen wir **ratio1** (Speichern nicht vergessen.)



A0, A1 and A2 are free moveable. Moving point B0 results in a simultaneous movement of B1.

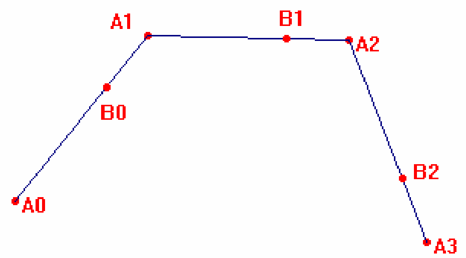
We define the whole contruction as a macro with A0, A1, B0 and A2 as initial objects and B1 as final object.

*We name the macro **ratio1**. Don´t forget to save the macro.*

Das Makro **ratio1** wird jetzt benutzt, um auf der Strecke A2A3 den Punkt B2 zu konstruieren.

Dabei müssen die Startobjekte in der richtigen Reihenfolge angeklickt werden:

A1, A2 , B1 , A3 und schon ist der Punkt B2 konstruiert.



*Applying **ratio1** we produce point B2 on the attached segment A2A3. Take care to click on the the initial objects in the right order: A1, A2, B1, A3 – and here we are: point B2 appears.*

Nun sollen auch auf den Strecken B0B1 und B1B2 die entsprechenden Teilungspunkte konstruiert werden. Dazu benötigen wir ein weiteres **Makro ratio2**, das den Teilungspunkt konstruiert, wenn die Strecken nicht „aneinanderhängen".

Wir nutzen dabei das letzte Bild – Startobjekte A0, A1, B0, A2, A3 Zielobjekt B2

In the next step we draw the segments B0B1 and B1B2 and find the division points C0 and C1.

*To do so we need another **macro ratio2**, which transfers the ratio on segments which are not attached.*

Initial objects are A0, A1, B0, A2, A3 and final object is B2.

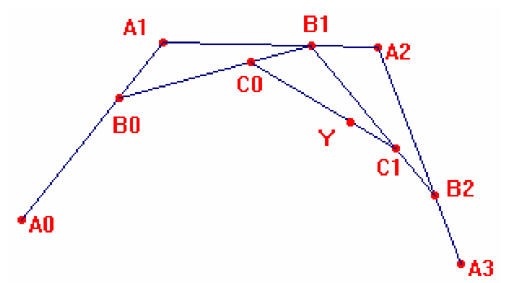
Auf diese Art konstruieren wir die Teilungspunkte B2 , C0 , C1 und schliesslich Y.

Wenn wir an B0 ziehen, verändert sich die ganze Konstruktion!!

Unser Augenmerk gilt dem Punkt Y.

Moving B0 results in a general movement of the whole figure.

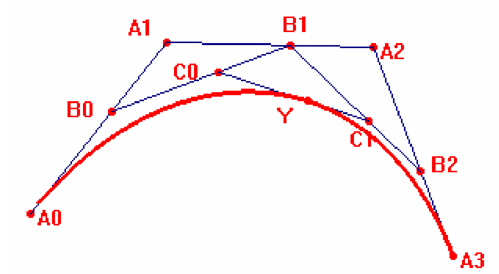
We have special interest in point Y.



Auf welcher Kurve bewegt sich der Punkt Y, wenn der Punkt B0 auf der Strecke A0A1 bewegt wird?

Diese **Ortskurve** lassen wir uns von CABRI einzeichnen und verstecken alle bei der Konstruktion benutzten Objekte.

Which is the ***locus*** *of Y when moving B0 on segment A0A1? We hide all auxiliary objects.*

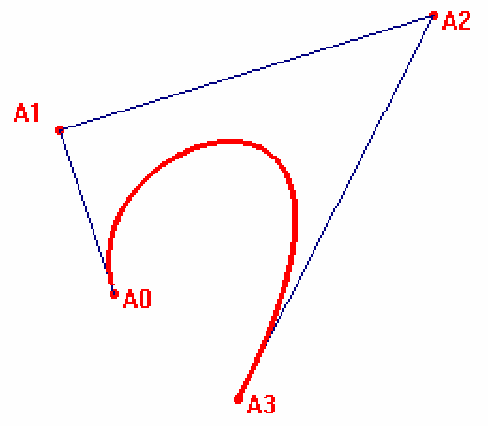


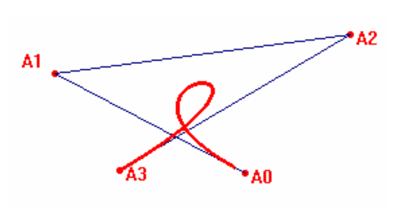
Nur die Punkte A0, A1, A2, A3 sind für die Form dieser Kurve verantwortlich.

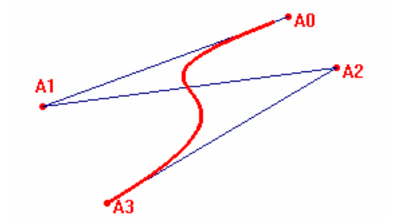
Zu guter Letzt definieren wir das Makro **cubbez** mit den Startobjekten A0, A1, A2, A3 und der Ortskurve als Zielobjekt.

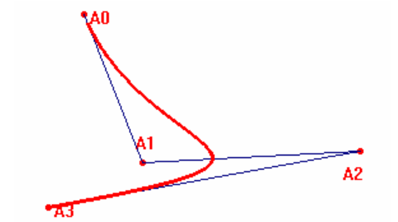
Only points A0, A1, A2 and A3 are responsible for the shape of this curve.

Finally we define the macro ***cubbez*** *with initial objects A0, A1, A2, A3 and the locus being the final object.*







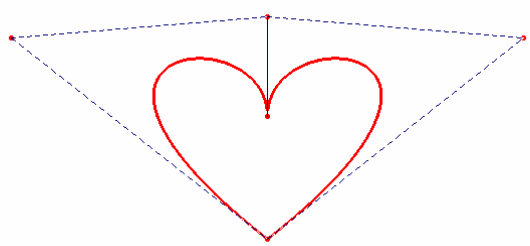


Bis hierher haben wir schon – CABRI sei Dank – ein schönes Stück Arbeit geleistet!

Wir haben dem Designer ein Werkzeug zur Verfügung gestellt, mit dem er schöne Kurven erzeugen kann – mit den mathematischen Grundlagen muss er sich nicht mehr auseinandersetzen. Wir erklären ihm, wie er das Werkzeug zu bedienen hat und schon kann er schöne Formen schaffen!

Thanks CABRI we did a good job. We have provided a tool for the designer which he or she can use for creating pretty curves without dealing with the mathematics.

We explain how to use the tool and he/she can work according to their creativity.



Aber wie kommen wir weiter?

Eigentlich möchten wir doch die Kurven auch mit Hilfe von Formeln beschreiben, damit wir genügend viele genaue numerische Daten beschaffen können. Wenn wir diese haben, können die Formen mit CNC-Maschinen hergestellt und beliebig oft kopiert werden. Ausserdem könnten wir die Daten zum Beispiel auf einer Diskette, einer CD oder via Internet an einen anderen Ort transportieren.

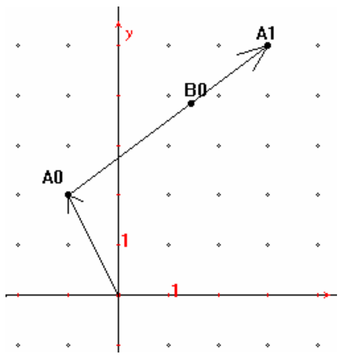
But how to proceed?

We would like to describe these curves by formulae in order to get arbitrary many accurate numerical data. Then we can use CNC-automatic machines to produce the shapes and copy them. Moreover the data can be stored an a diskette or CD and/or transmitted easily by Internet. We will focus on this problem now.

Mit diesem Problem wollen wir uns jetzt beschäftigen.

3. Die algebraische Beschreibung *The algebraical description*

(Ein Häppchen Vektorrechnung)



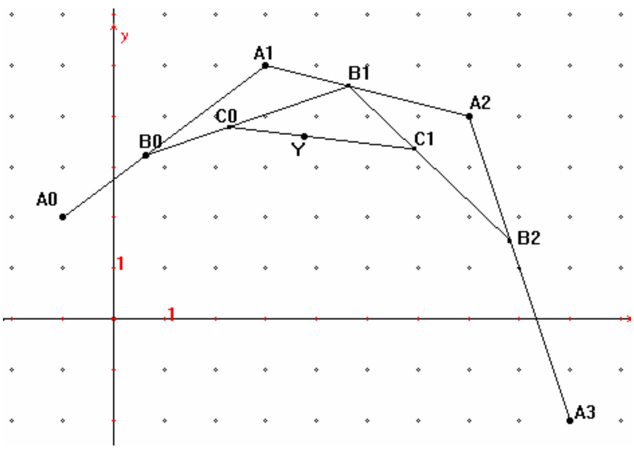
Erinnern wir uns an unsere Ausgangssituation:

Der Punkt B0 liegt auf der Strecke A0A1, das können wir so aufschreiben:

B0 = A0 + t·(A1 – A0)

B0 = A0 + t·A1 – t·A0

B0 = (1 – t)·A0 + t·A1 (dabei ist $0 \le t \le 1$)



Auf Grund unserer Konstruktion gilt diese Gleichung auch bei den anderen Punkten:

B1 = (1-t)·A1+t·A2

B2 = (1-t)·A2+t·A3

und

C0 = (1-t)·B0+t·B1

C1 = (1-t)·B1+t·B2

und schließlich

Y = (1-t)·C0+t·C1

Durch Einsetzen und Umstellen der Formeln ergibt sich:

$$\boxed{Y = \sum_{k=0}^{3} \binom{3}{k} \cdot t^k \cdot (1-t)^{3-k} \cdot Ak}$$

Das sind die **kubischen Bézierkurven**.

Substitution and rearranging leads to the sum-expression given in the box. These are ***Cubic Beziér Curves****. We can do the manipulation by hand or leave it to DERIVE.*

#1: $[A0 :=,\ A1 :=,\ A2 :=,\ A3 :=]$

#2: $[B0 := (1 - t)\cdot A0 + t\cdot A1,\ B1 := (1 - t)\cdot A1 + t\cdot A2,\ B2 := (1 - t)\cdot A2 + t\cdot A3]$

#3: $[C0 := (1 - t)\cdot B0 + t\cdot B1,\ C1 := (1 - t)\cdot B1 + t\cdot B2]$

#4: $Y := (1 - t)\cdot C0 + t\cdot C1$

#5: $Y := A0\cdot(1 - t)^3 + t\cdot(3\cdot A1\cdot(t - 1)^2 - t\cdot(3\cdot A2\cdot(t - 1) - A3\cdot t))$

Applying Expand on Subexpressions leads to:

#6: $Y := A0\cdot(1 - t)^3 + (3\cdot A1\cdot t\cdot(t - 1)^2 - t^2\cdot(3\cdot A2\cdot(t - 1) - A3\cdot t))$

#7: $Y := A0\cdot(1 - t)^3 + (3\cdot A1\cdot t\cdot(t - 1)^2 - (3\cdot A2\cdot t^2\cdot(t - 1) - A3\cdot t^3))$

which is equal to

#8: $Y := A0\cdot(1 - t)^3 + 3\cdot A1\cdot t\cdot(t - 1)^2 - 3\cdot A2\cdot t^2\cdot(t - 1) + A3\cdot t^3$

Bei der Darstellung der Bézierkurven werden die sogenannten **Bernsteinpolynome**

$$B_k^n(t) = \binom{n}{k}\cdot t^k \cdot (1-t)^{n-k} \qquad (*)$$

mit den Gewichtungsfaktoren Ak multipliziert und addiert. Diese Polynome hat der Mathematiker **S.N.Bernstein im Jahr 1911** für einen konstruktiven Beweis des **Weierstraßschen Approximationssatzes** eingeführt.

*For presentation of Bézier Curves the so called **Bernstein Polynomials** (*) are multiplied by weight factors Ak and added. **S.N.Bernstein** introduced these polynomials 1911 for a constructive proof of the **Weierstraß Approximation Theorem**.*

Wir wählen z.B. n=5 und lassen alle 6 Bernsteinpolynome zeichnen.

Welche Eigenschaften haben sie?

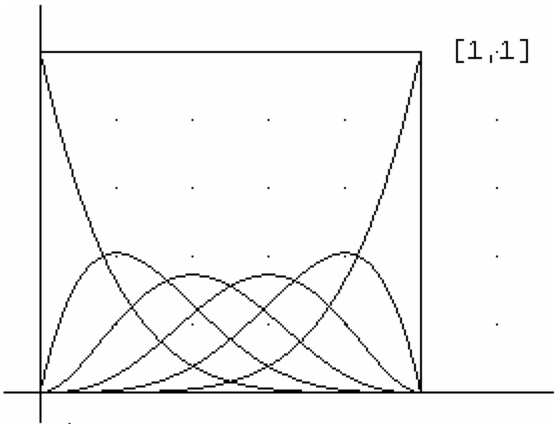
Wo liegen die Extremwerte?

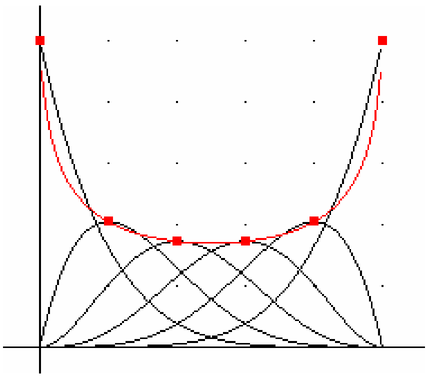
We choose n=5 and plot all of the six Bernstein polynomials.

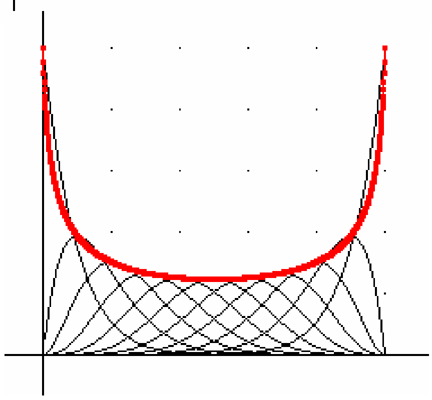
What do they have in common?

Find the locus of the extremal points!

Hint: If you cannot find the locus, then open the file bernstein.dfw.







Die Herleitung der Ortslinie der Extremwerte finden Sie in bernstein.dfw.

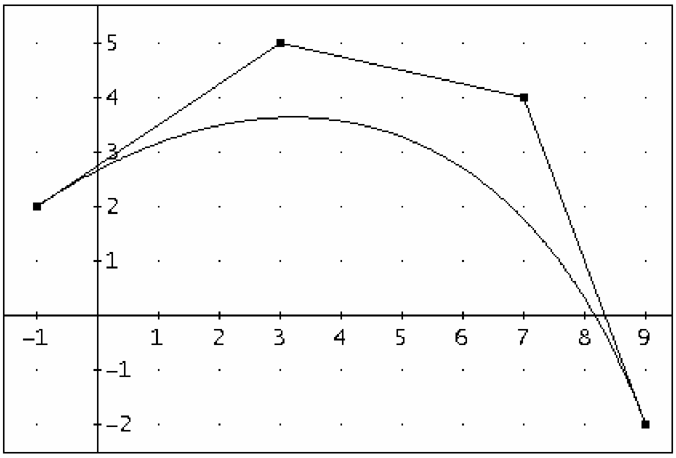
Die oben definierten Bézierkurven können mit Derive direkt gezeichnet werden. Wir müssen nur die Koordinaten der 4 Punkte kennen (diese lesen wir aus der CABRI-Zeichnung ab).

#9: $\text{cub\_bez}(p0, p1, p2, p3, t) := p0\cdot(1-t)^3 + 3\cdot p1\cdot t\cdot(t-1)^2 - 3\cdot p2\cdot t^2\cdot(t-1) + p3\cdot t^3$

#10: [A0 := [-1, 2], A1 := [3, 5], A2 := [7, 4], A3 := [9, -2]]

#11: Y(t) := cub_bez(A0, A1, A2, A3)

#12: $Y(t) := \left[-2\cdot t^3 + 12\cdot t - 1,\ -t^3 - 12\cdot t^2 + 9\cdot t + 2\right]$



4. Bézierkurven im Raum *Bézier Curves in Space*

Die bis jetzt erworbenen Erkenntnisse für die 2D-Grafik übertragen wir nun auf den Raum. Wir wählen 4 Punkte und zeichnen den Streckenzug in der 3D-Grafik. Danach definieren wir unsere Bézierkurve wie im 2-dimensionalen Fall und lassen diese zeichnen:

We transfer all the knowledge acquired so far into space. We choose 4 points, draw the segments, define the space curve using the function which we have just defined and plot the curve. Don´t forget to set the domain for the parameter from 0 to 1 (via Insert > Plot)

Bézier Curves in Space

#13: [S0 := [4, 0, 5], S1 := [0, 7, 0], S2 := [3, 2, 9], S3 := [-2, 0, 2]]

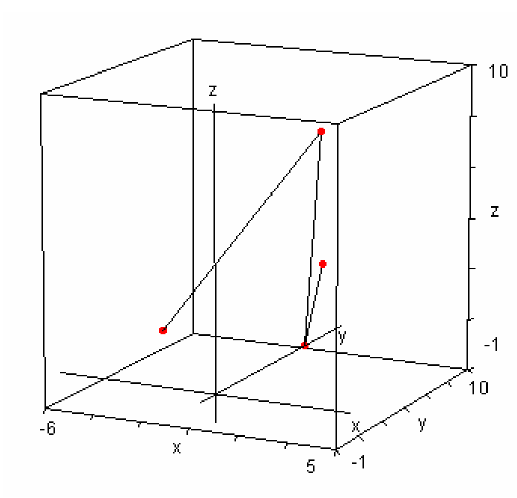
#14: $\left[[S0, S1, S2, S3], \begin{bmatrix} S0 \\ S1 \\ S2 \\ S3 \end{bmatrix}\right]$

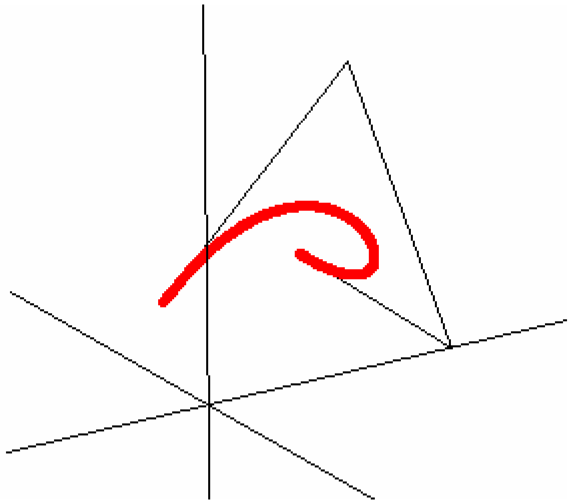
#15 results in the space curve and #17 results in the space curve plotted as a thick line (Insert > Plot > Point Size Small or Medium)

#15: cub_bez(S0, S1, S2, S3)

#16: $\left[-15\cdot t^3 + 21\cdot t^2 - 12\cdot t + 4,\ 3\cdot t\cdot(t-1)\cdot(5\cdot t - 7),\ -30\cdot t^3 + 42\cdot t^2 - 15\cdot t + 5\right]$

#17: (TABLE([[cub_bez(S0, S1, S2, S3)]], t, 0, 1, 0.002))↓↓2

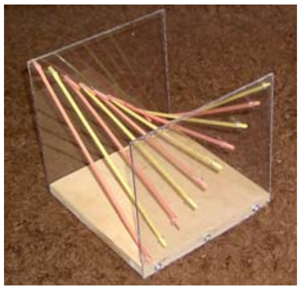


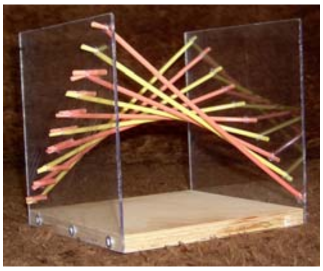


5. Die Idee der HP-Flächen zur Erzeugung schöner Flächen

The concept of HP-Surfaces for designing beautiful surfaces

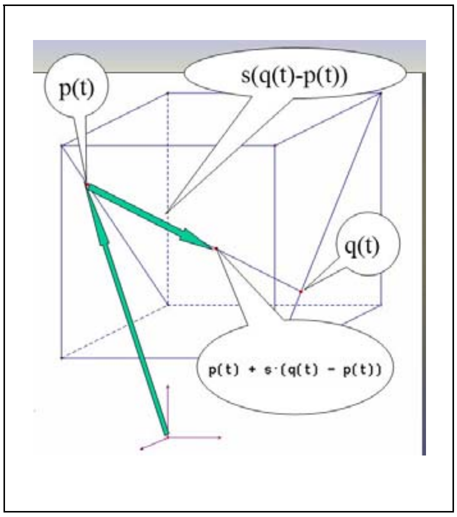
Wenn in einem Würfel die Punkte gegenüberliegender Diagonalen so wie in den Bildern dargestellt durch Geraden verbunden werden, dann entsteht eine Sattelfläche – das gegensinnig gekrümmte Hyperboloid (die HP-Fläche).





Diese Flächen lassen sich im 3D-Fenster von Derive ganz einfach darstellen, wenn man den Erzeugungs-prozess zur Beschreibung mit einer Formel nutzt.

These surfaces are very easy to define and to present as well with DERIVE. We define the vertices of the cube, create the sequences of points on the two diagonals of opposite sides of the cube and find then the connecting segments of respective points of these diagonals:



The HP-Surface

#18: [a := [2, 0, 0], b := [2, 2, 0], c := [0, 2, 0], d := [0, 0, 0]]

#19: [e := [2, 0, 2], f := [2, 2, 2], g := [0, 2, 2], h := [0, 0, 2]]

#20: [a, b, c, d, a, e, f, b, f, g, c, g, h, d, h, e]

The points on the opposite diagonals:

#21: p(t) := e + t·(d − e)

#22: q(t) := b + t·(g − b)

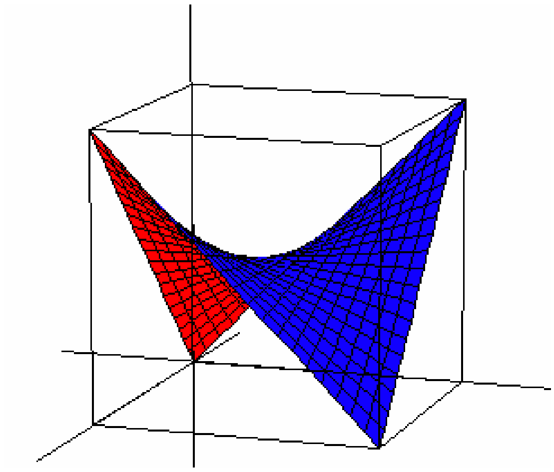
The point on the surface = parameter representation $(0 \le t, s \le 1)$

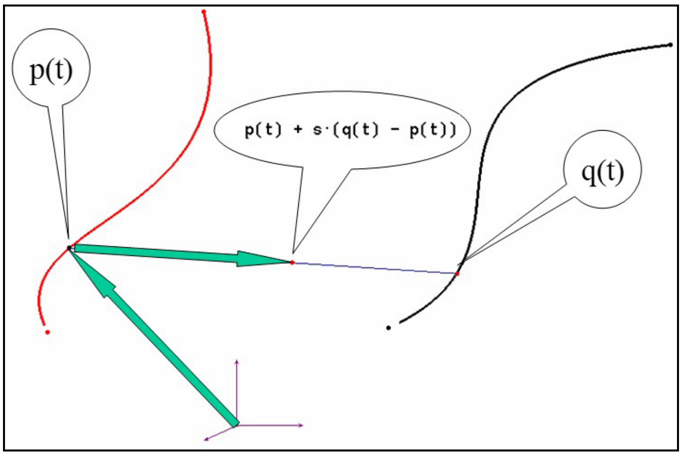
#23: hp(t, s) := p(t) + s·(q(t) − p(t))

#24: hp(t, s) := [2 − 2·t, 2·s, 2·s·(2·t − 1) − 2·t + 2]

Diese Flächen spielen in der Architektur bei der Gestaltung von ungewöhnlichen Dachflächen eine Rolle, da sie durch eine Schalung mit geraden Brettern in Stahlbetonbauweise leicht zu realisieren sind.

These surfaces play an important role in architecture for designing unusual roofs, because they are easy to realise in an armored concrete construction using straight logs for the encasing.

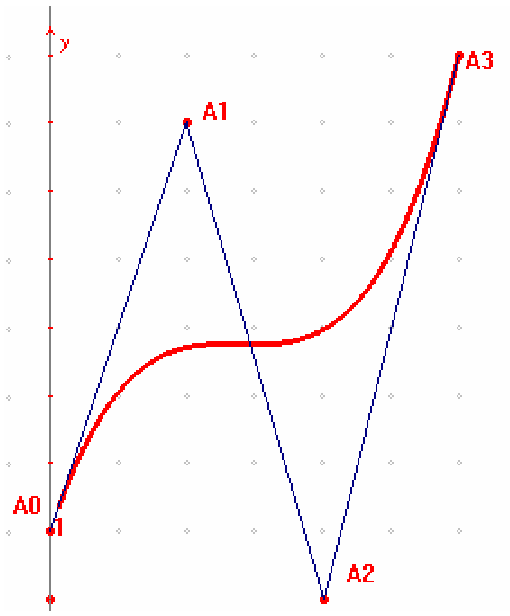


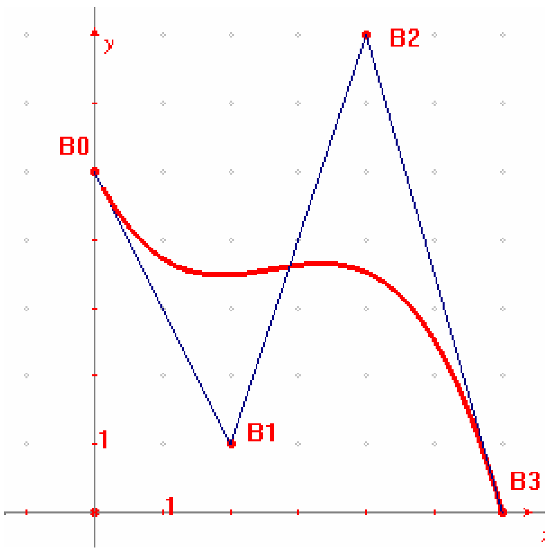


Wir wollen nun die Idee der Erzeugung einer HP-Fläche erweitern, indem wir P(t) und Q(t) nicht mehr auf Strecken, sondern auf Bézierkurven im Raum bewegen.

We will extend the idea of creating a HP-surface by moving P(t) and Q(t) not on straight lines but on Bézier curves in 3D space.

Let´s design two "beautiful" boundary curves with the DGS.





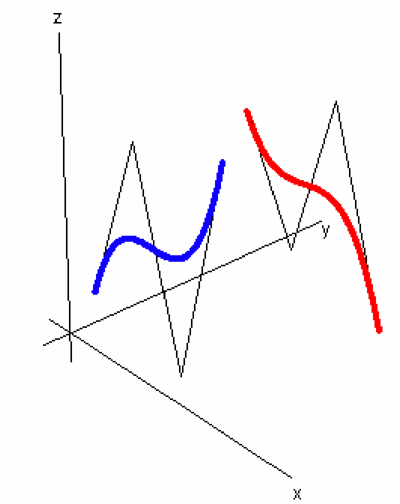
Diese transportieren wir in den Raum, indem wir die Koordinaten der Steuerpunkte als x- und z-Koordinaten verwenden. A0 bis A3 setzen wir an die Stelle y=1 und B0 bis B3 setzen wir an die Stelle y=7 und beschreiben die Randkurven als kubische Bézierkurven und zeichnen sie mit Derive …

We convert the two curves into space by using the coodinates of the control points as x- and z-coordinates. In A0 through A3 we set y = 1 and in B0 through B3 we set y = 7 and describe the border curves as cubic Bézier curves and plot them with Derive …

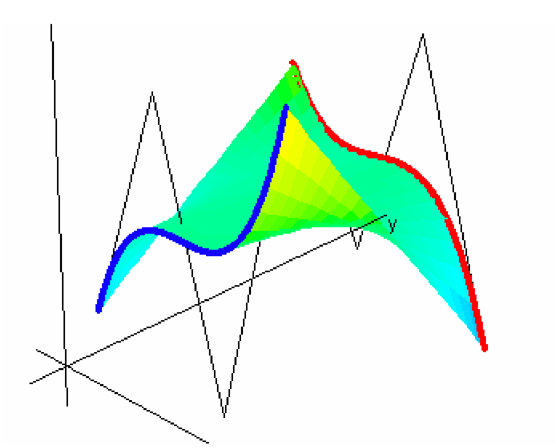
```
[A0 := [0, 1, 1], A1 := [2, 1, 7], A2 := [4, 1, 0], A3 := [6, 1, 8]]

[B0 := [0, 7, 5], B1 := [2, 7, 1], B2 := [4, 7, 7], B3 := [6, 7, 0]]

[p(t) := cub_bez(A0, A1, A2, A3), q(t) := cub_bez(B0, B1, B2, B3)]

⎡ [A0, A1, A2, A3] ⎤
⎣ [B0, B1, B2, B3] ⎦

(TABLE([[p(t)]], t, 0, 1, 0.002))↓↓2

(TABLE([[q(t)]], t, 0, 1, 0.002))↓↓2
```

… und schließlich werden die Randpunkte durch Strecken verbunden.

… and finally we connect the boundary points by segments.



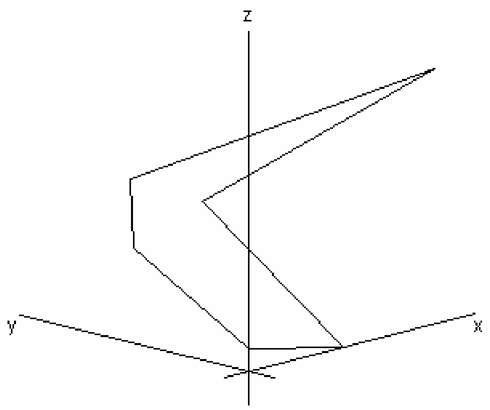

```
y(t) := p(t) + s·(q(t) - p(t))
```

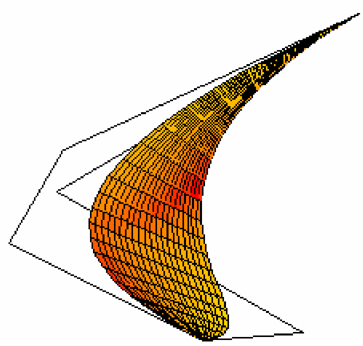


Auf diese Art und Weise lassen sich schöne Flächen erzeugen, zB kann man beide Kurven im Raum an den gleichen Punkten beginnen und enden lassen.

In this way we can create pretty surface: e.g. let both curves begin and end in common points.

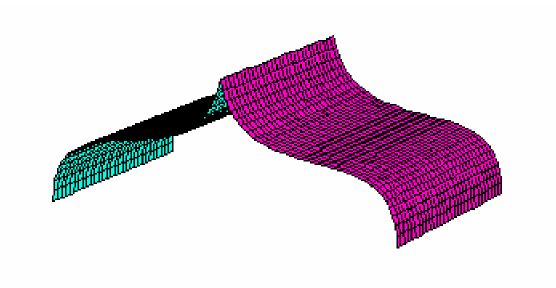
```
[U0 := [2, 2, 0], U1 := [4, 0, 0], U2 := [6, 8, 3], U3 := [8, 0, 8]]

[V0 := [2, 2, 0], V1 := [3, 8, 2], V2 := [0, 5, 5], V3 := [8, 0, 8]]

⎡ [U0, U1, U2, U3] ⎤
⎣ [V0, V1, V2, V3] ⎦

[p_(t) := cub_bez(U0, U1, U2, U3), q_(t) := cub_bez(V0, V1, V2, V3)]

p_(t) + s·(q_(t) - p_(t))
```





Oder wir können ein Dach gestalten.

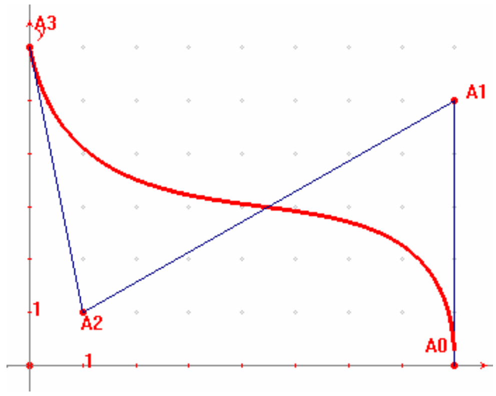
Or we can design a roof.



6. Gestaltung von Rotationsflächen *Designing of Solids of Revolution*

Die Kontur der Fläche entwerfen wir zunächst mit unserem DGS-Werkzeug:

First of all we create the profile curve of the surface using our DGS-tool:



Diese wird auf die erste Achse gesetzt ([x,y] → [x,0,y]) und dann im Raum gezeichnet:

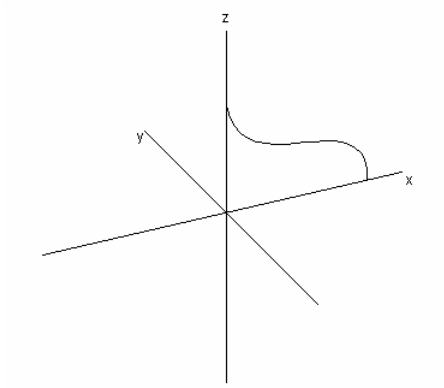
The curve is placed on the 1st axis ([x,y] → [x,0,y]) *and then plotted in space*:

Solid of revolution

#14: [W0 := [8, 0, 0], W1 := [8, 0, 5], W2 := [1, 0, 1], W3 := [0, 0, 6]]

#15: w(t) := cub_bez(W0, W1, W2, W3)

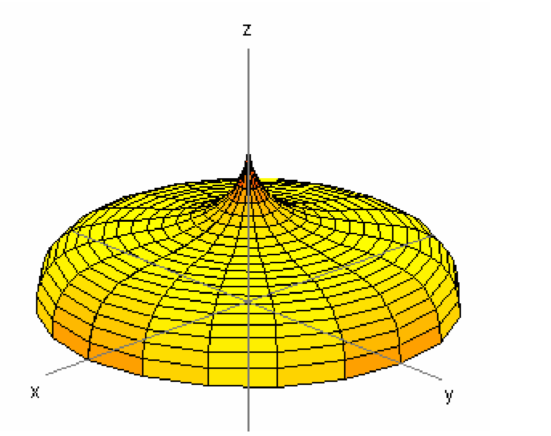
#16: $w(t) := \left[(t-1)\cdot(13\cdot t^2 - 8\cdot t - 8),\ 0,\ 3\cdot t\cdot(6\cdot t^2 - 9\cdot t + 5)\right]$

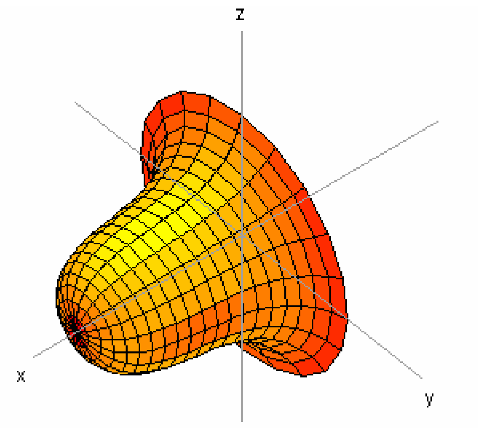


Schließlich wird diese Kurve erst um die z-Achse und dann um die x-Achse gedreht (Drehwinkel s läuft jeweils von 0 bis 2π).

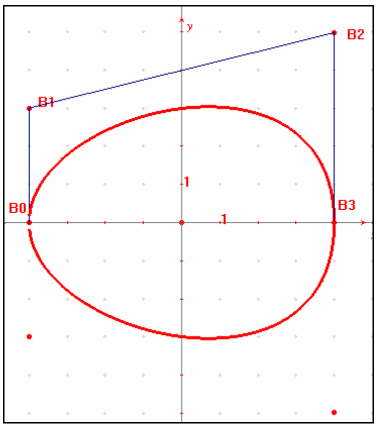
Finally we rotate this curve around the z-axis (first) and around the x-axis (later). (rotation angle s is running from 0 to 2π).

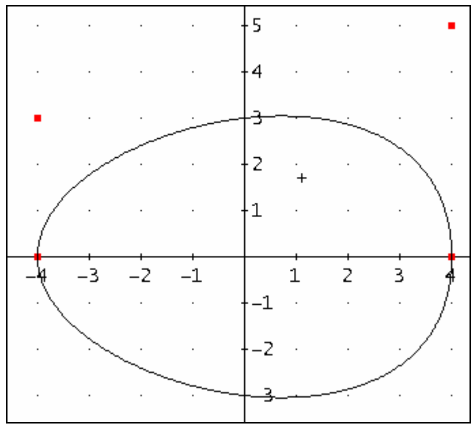
$\left[(w(t))_1\cdot \mathrm{COS}(s),\ (w(t))_1\cdot \mathrm{SIN}(s),\ (w(t))_3\right]$ $\left[(w(t))_1,\ (w(t))_3\cdot \mathrm{COS}(s),\ (w(t))_3\cdot \mathrm{SIN}(s)\right]$





Oder wir entwerfen ein Ei, das in der 3D-Graphik dargestellt werden soll. Zunächst entwerfen wir mit CABRI einen Querschnitt, mit dem wir zufrieden sind, entnehmen die Koordinaten der Steuerpunkte der CABRI-Zeichnung und stellen den Querschnitt im 2D-Fenster dar (rechtes Bild): *(Let´s boil a Cubic Bézier Egg)*





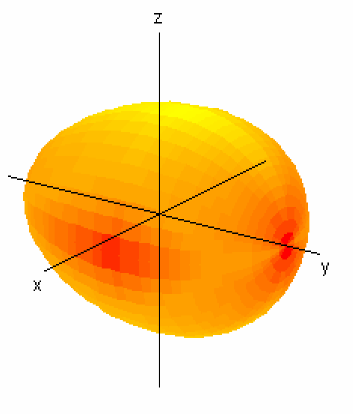
$$[E0 := [-4,\ 0],\ E1 := [-4,\ 3],\ E2 := [4,\ 5],\ E3 := [4,\ 0]]$$

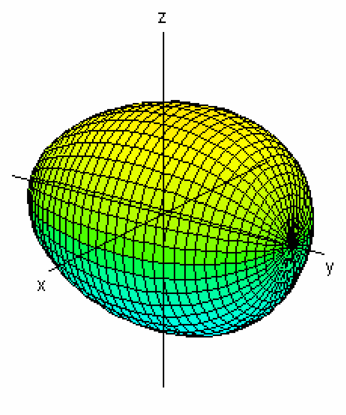
$$ei(t) := cub\_bez(E0,\ E1,\ E2,\ E3)$$

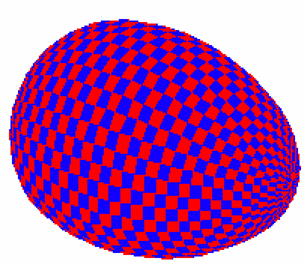
$$ei(t) := \left[-4\cdot(4\cdot t^3 - 6\cdot t^2 + 1),\ 3\cdot t\cdot(1 - t)\cdot(2\cdot t + 3) \right]$$

Diese Kurve lassen wir im 3D-Fenster um die y-Achse rotieren:

$$\left[(ei(t))_2\cdot COS(s),\ (ei(t))_1,\ (ei(t))_2\cdot SIN(s) \right]$$







7. Bézierflächen *Bézier Surfaces*

Zu guter Letzt erzeugen wir noch Bézierflächen über rechteckigen Parametergebieten. Dabei wird nicht mehr nur eine Punktfolge definiert, in die eine Kurve „gehängt" wird, sondern wir benutzen ein ganzes Netz aus Punkten, das wir zur Erzeugung einer Fläche nutzen.

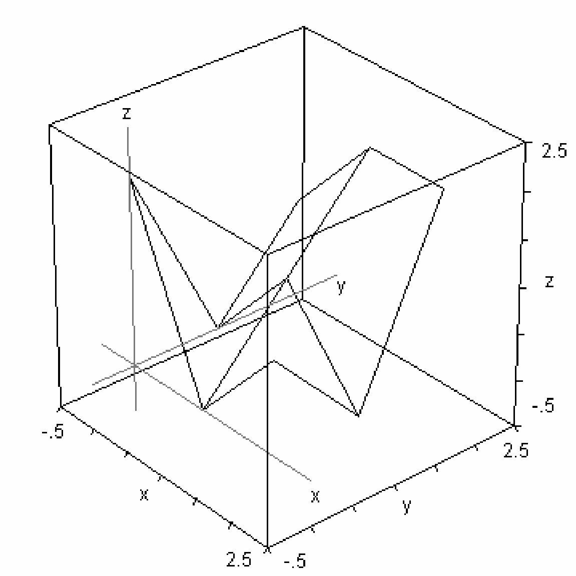
Zunächst definieren wir neun Steuerpunkte:

```
[A0 := [0, 0, 2], B0 := [1, 0, 0], C0 := [2, 0, 1]]

[A1 := [0, 1, 0], B1 := [1, 1, 1], C1 := [2, 1, 0]]

[A2 := [0, 2, 1], B2 := [1, 2, 2], C2 := [2, 2, 2]]
```

Damit können wir das Netz der entstehenden Fläche zeichnen:

In the last paragraph we create a Bézier surface above a rectangle parameter domain. We don´t define a sequence of points, but a "net of points". Nine control points are defined and the net can be plotted:

$$\begin{bmatrix} [A0, A1, A2] \\ [B0, B1, B2] \\ [C0, C1, C2] \\ [A0, B0, C0] \\ [A1, B1, C1] \\ [A2, B2, C2] \end{bmatrix}$$



Wir benötigen die Bernsteinpolynome und definieren unsere Matrix der Steuerpunkte unter dem Namen `dat`, damit wir auf die einzelnen Punkte in der folgenden Formel zurückgreifen können:

We need the Bernstein polynomials and define a matrix `dat` of the control points in order to refer to the single points applying the respective formula:

$$dat := \begin{bmatrix} A0 & B0 & C0 \\ A1 & B1 & C1 \\ A2 & B2 & C2 \end{bmatrix}$$

$$bern(n, k, p) := COMB(n, k) \cdot p^{k} \cdot (1 - p)^{n-k}$$

Die zugrunde liegende Idee:

Wir erzeugen zeilenweise Bézierkurven (mit dem Parameter *t*), die dann spaltenweise durch Bernsteinpolynome (mit dem Parameter *s*) verbunden werden.

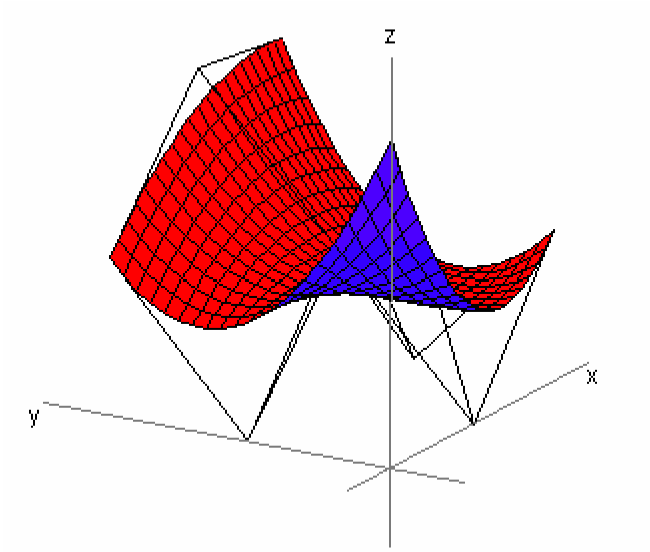
The idea is to create Bézier Curves (parameter t) for the rows, which are connected by Bernsteinpolynomials in the columns (parameter s).

$$\sum_{i=0}^{2} bern(2, i, s) \cdot \sum_{j=0}^{2} dat_{i+1,j+1} \cdot bern(2, j, t)$$

or $(0 \le s,t \le 1)$

$$\sum_{i=0}^{2} \sum_{j=0}^{2} dat_{i+1,j+1} \cdot bern(2, i, s) \cdot bern(2, j, t)$$

Schließlich wird die Fläche gezeichnet:



Jetzt kommt die Krönung:

Um zu untersuchen, wie die Form dieser Fläche durch Variation der Steuerpunkte verändert wird, benutzen wir Schieberegler und verändern die Fläche dynamisch. Wir variieren zB die 3. Koordinate des mittleren Punktes (Variable z) und führen für z einen Schieberegler ein mit -3 ≤ z ≤ 5 und 32 Intervallen.

We will finish with the highlight:

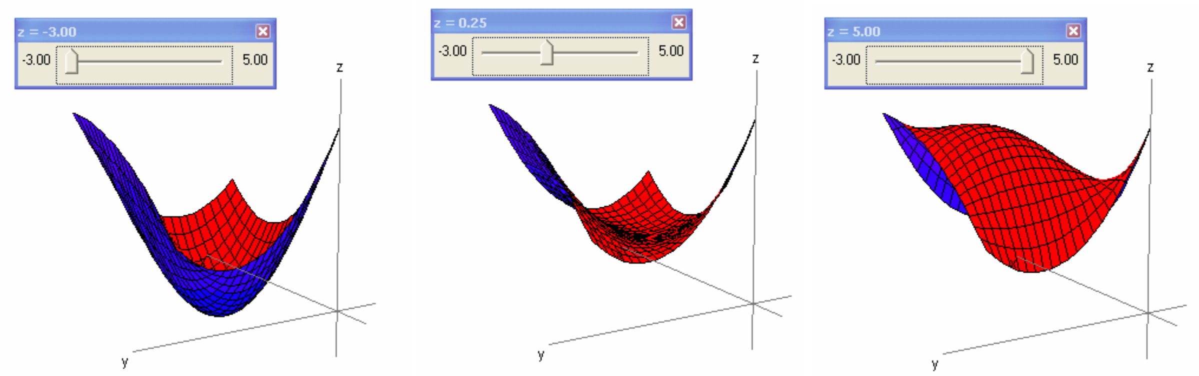
In order to investigate the influence of the control points on the shape of the surface we introduce one (or even more) slider bars and vary the surface dynamically. So we vary for example the 3rd coordinate of the "center point" B1 and insert a slider bar for z with -3 ≤ z ≤ 5 and 32 intervals.

$$B1\_ := [1,\ 1,\ z]$$

$$dat2 := \begin{bmatrix} A0 & B0 & C0 \\ A1 & B1\_ & C1 \\ A2 & B2 & C2 \end{bmatrix}$$

$$\sum_{i=0}^{2} bern(2,\ i,\ s)\cdot \sum_{j=0}^{2} dat2_{i+1,\,j+1}\cdot bern(2,\ j,\ t)$$

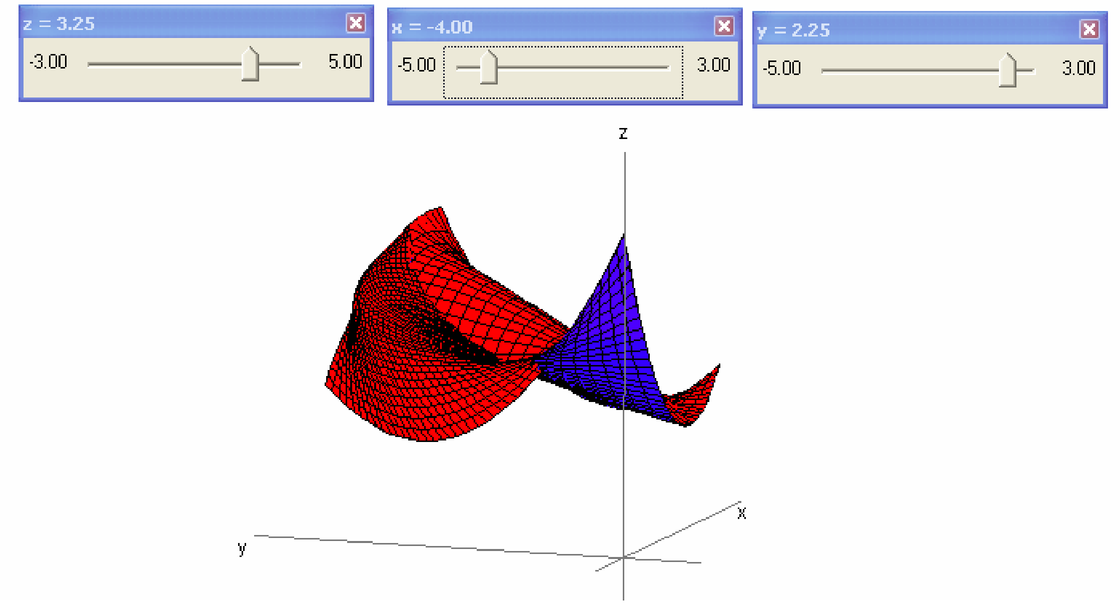
$$\left[2\cdot t,\ 2\cdot s,\ 4\cdot s\cdot t\cdot z\cdot(s-1)\cdot(t-1) + s^2\cdot(2\cdot t^2 - 2\cdot t + 3) - 2\cdot s\cdot(3\cdot t^2 - 4\cdot t + 2) + 3\cdot t^2 - 4\cdot t + 2\right]$$



Der Phantasie und Kreativität zur Gestaltung von schönen Formen sind keine Grenzen gesetzt!

There are no limits for phantasy and creativity to design beautiful shapes!

I took Hubert´s closing sentence for serious and introduced two more slider bars for the 1st and 2nd coordinate of control point B1. Josef



References:

[1] *Grabinger, Benno*, Projekte und Aufgaben zur Analytischen Geometrie, Schroedel, Hannover, 1999

[2] *Kleifeld, Achim*, Geometrisches Modellieren mit Bézierkurven verbindet anwendungsbezogen Analysis, Lineare Algebra und Algorithmik, In: Förster,Henn,Meyer (Hrsg), Materialien für einen realitätsbezogenen Mathematikunterricht, Band 6 Computeranwendungen S.61-79, Franzbecker, Hildesheim, 2000

[3] *Meyer, Jörg*, Bezierkurven In: Förster,Henn,Meyer (Hrsg), Materialien für einen realitätsbezogenen Mathematikunterricht, Band 6 Computeranwendungen S.44-60, Franzbecker, Hildesheim, 2000

[4] *Bungartz, H.J, Griebel, M., Zenger,C.*, Einführung in die Computergraphik, Vieweg, Braunschweig, 2002

[5] Lexikon der Mathematik, Band 1, Spektrum Akademischer Verlag, Heidelberg, 2001

Anschrift des Autors:

Dr. Hubert Weller
Vogelsang 10
D-35633 Lahnau
Germany

Earlier DNL-Contributions dealing with Bézier Curves and HP-surfaces:

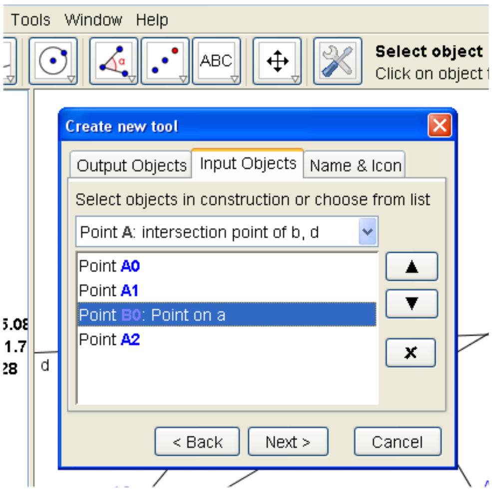
[6] *Günter Scheu*, An Approach to the Bézier Curves, DNL#19 (1955)

[7] *Franz Schlöglhofer*, Bézier Curves in School, DNL#52, (2003)

[8] *Benno Grabinger*, A Mathematical Potato Chip, DNL#59 (2005)

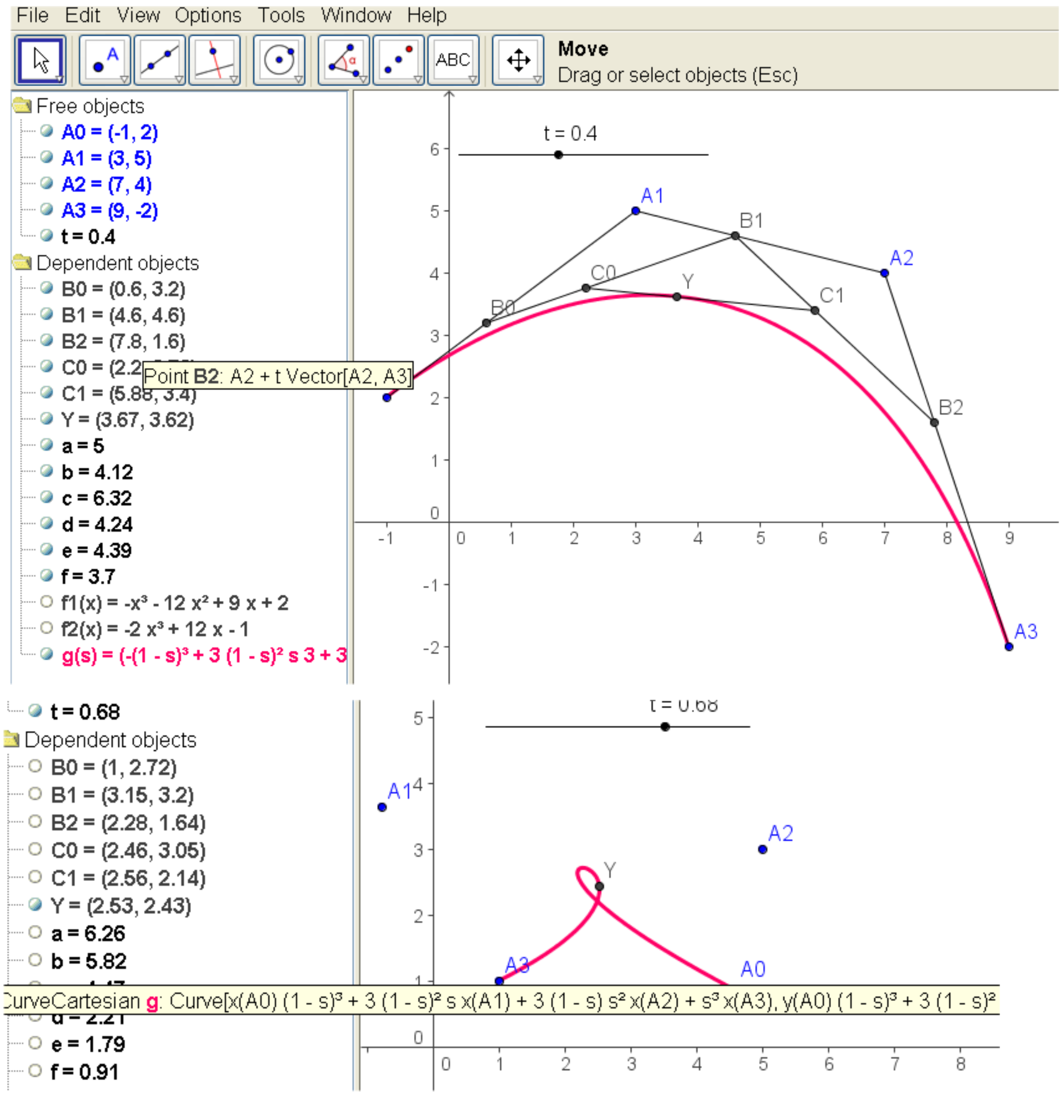
As I announced earlier here are some screen shots from Bézier Curves & GeoGebra. Josef.

We can create MACROs (see one step in designing the first macro from above).



The picture below shows that we can enter the position vectors of A0 through Y directly - according to the calculation on page 11. Parameter t is animated by a slider bar. (See the DERIVE procedure in [7]).

The pre-release version of GeoGebra enables the user to enter the cubic Bézier curve in its parameter form (calculated supported by DERIVE). According to the philosophy of a DGS we can animate this respresentation by draging any point and the respective algebraic representation is given in the „Algebra Window“ simultaneously. (Last screen shot on this page)



You can see the algebraic form of the locus (with parameter s, because t is reserved for the slider bar) which can be entered in the Edit line. It would be great if CABRI could implement similar tools to narrow the gap between geometric and algebraic representation forms.

Step Functions and Riemann's Concept of Integration

by Wolfgang Pröpper, Nürnberg, Germany

1. Step Functions[1]

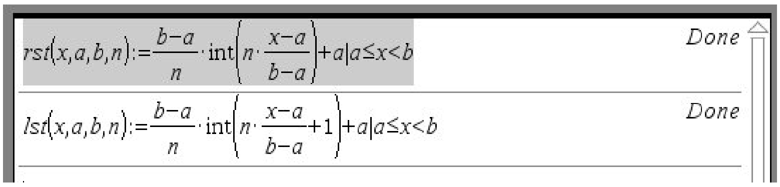
The expression $\frac{x-a}{b-a}$ maps all x $\in$ [a .. b[onto [0 .. 1[. Therefore $n \cdot \frac{x-a}{b-a}$ takes values from [0 .. n[and finally $\text{int}\left(n \cdot \frac{x-a}{b-a}\right)$ is already a step function with integer values from 0 to n-1, while the interval [a .. b[is subdivided into n subintervals.

To change the height of the steps we multiply with (a so called step factor h =) $\frac{b-a}{n}$, add a and restrict the range to $a \le x < b$ and so we get

$$\text{rst}(x,a,b,n) := \frac{b-a}{n} \cdot \text{int}\left(n \cdot \frac{x-a}{b-a}\right) + a \mid a \le x < b$$

We call it "**r**ight **st**ep function" as you will see in a graphical representation in a few minutes. It takes values $a + k \cdot \frac{b-a}{n}$ in the kth subinterval of [a .. b[, when k runs from 0 to n-1.

Before plotting we define a 2nd function, which we call "**l**eft **st**ep function as can be seen at the screenshot to the right.



Its values are $a + (k+1) \cdot \frac{b-a}{n}$ (with k running from 0 to n-1) and the steps produced hereby show to the left. In G & G we define f1(x) = x first to show the impact of the just defined functions.

If we take a = 1, b = 6 and n = 4 for instance, we define

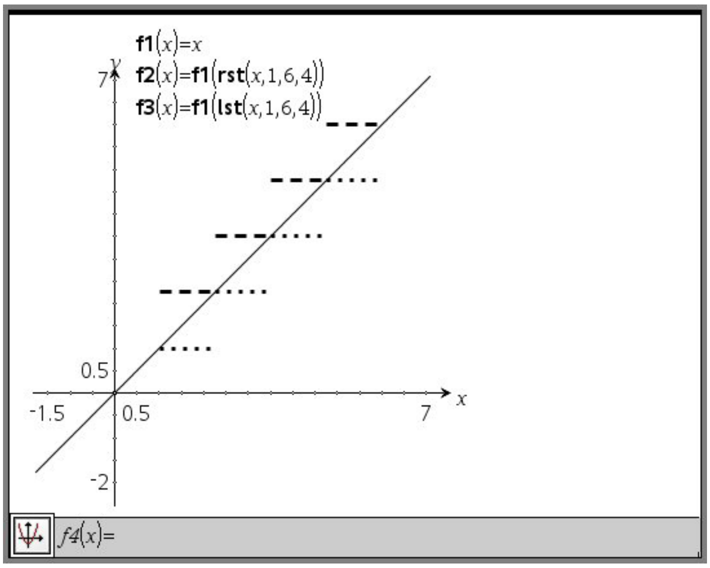
f2(x) = f1(rst(x,1,6,4))

and

f3(x) = f1(lst(x,1,6,4)).

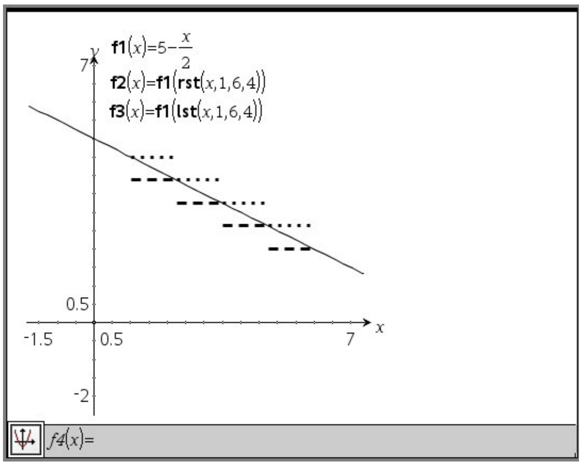
After some cosmetic treatment (oh, how I wish Nspire knew colors, at least in the PC-version!) we easily see the meaning of "right" and "left" steps.

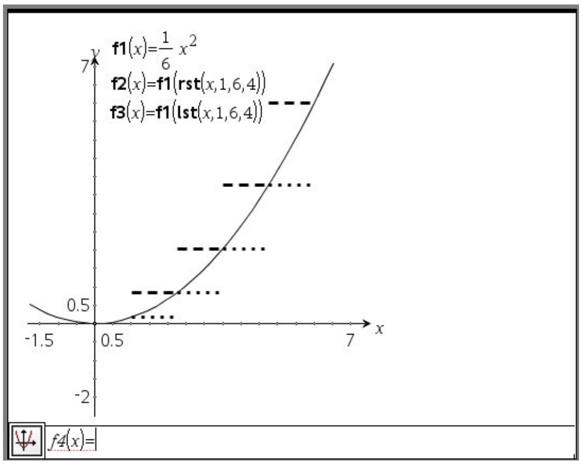
(The figure to the right puts in mind that right steps could also be regarded as lower steps and left steps as upper steps. But that is only valid for increasing functions.)



[1] I got the idea for handlig step functions this way from Philippe Fortin. Thank you, Philippe!

Now we have a tool to transfer the step concept to any function. We just have to change f1(x). For instance we take $f1(x) = 5 - \frac{x}{2}$ or $f1(x) = \frac{1}{6}x^2$ as can be seen in the next two figures.





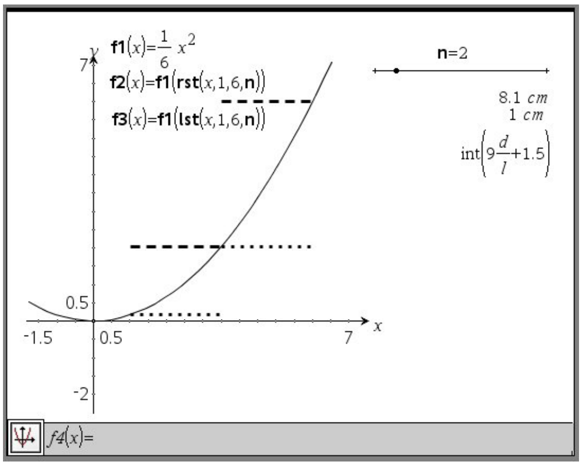
2. Building a Slide Control

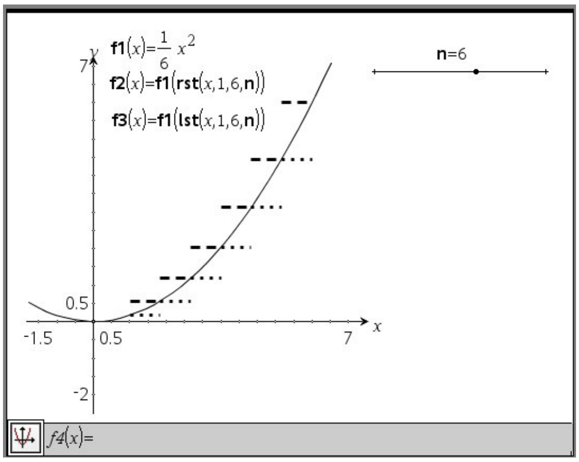
To get some more dynamics into the scene, we build an "integer slider bar".

For short:

Put a point on an arbitrary segment. Then measure the length of the segment and the distance of that point from the left edge of the segment. Type the expression $\text{int}\left(9 \cdot \frac{d}{l} + 1.5\right)$ (to get an integer slider from 1 to 10) as a text in the G & G screen above and let it be calculated (with d as distance and l as length). Assign the calculated value of this construction to a variable **n** and finally hide the expression togetherwith the two measurements.

Now we return to $f1(x) = \frac{1}{6}x^2$ and change f2(x) = f1(**rst**(x,1,6,**n**)) and f3(x) = f1(**lst**(x,1,6,**n**)) respectively. By moving the slider we see how the figure changes.





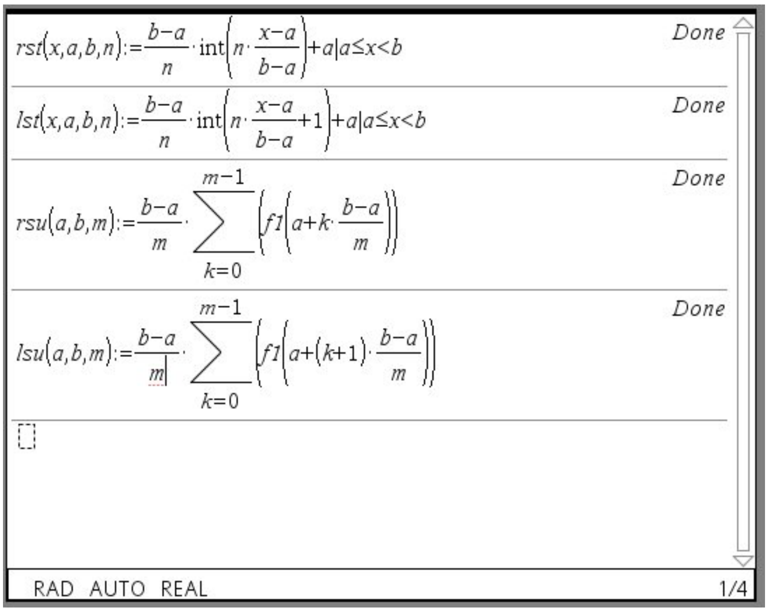
(In the left figure the slider construction is still visible.)

3. Riemann Sums

Now it would be a great thing if Nspire would have a shade feature to shade the area between the x-axis and the graph of a function (for instance our f2 and f3 functions). But as this concept does not yet exist (but I do hope it will come) we have to stress our imagination when we want to calculate the area between the x-axis and our step functions.

It is obvious that the width of the steps is $\frac{b-a}{n}$. And the height of the steps is $f1(a+k\cdot\frac{b-a}{n})$ for the right steps and $f1(a+(k+1)\cdot\frac{b-a}{n})$ for the left steps with k running from 0 to n-1.

So we define the two functions **rsu**(a,b,m) and **lsu**(a,b,m) (standing for **r**ight **su**m and **l**eft **su**m respectively) as can be seen at the right.



It would be very nice to type *rsu(1,6,n)* for instance as a text in our G & G screen and let it be calculated.

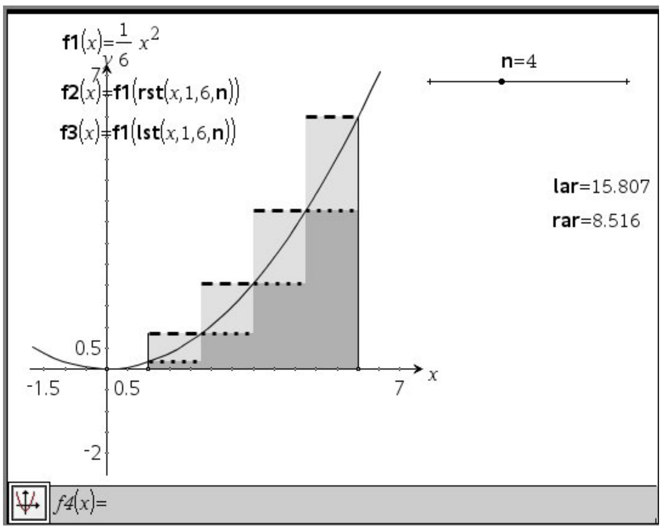
But now comes another bad news:

The calculate concept of Nspire does not work with self-defined functions like in this example. (But I also here hope the "not" is a "not yet".)

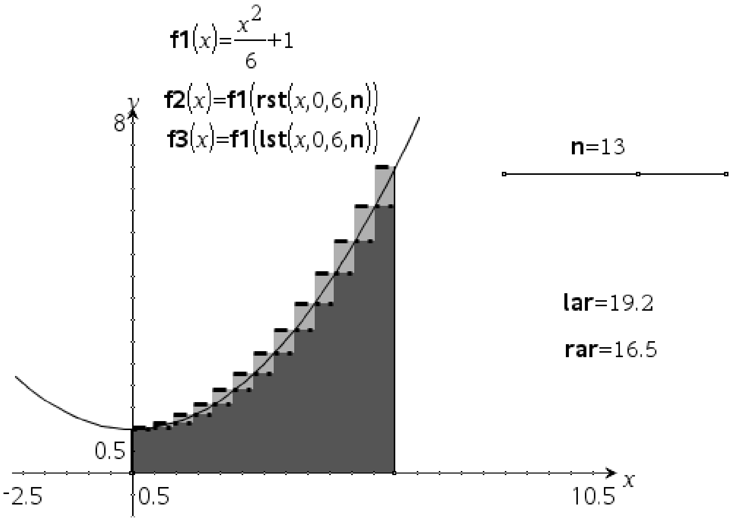
To get the areas despite of the missing calculate feature in a vivid manner, we use the integrate tool in G & G. I know, this is a point where, as a German proverb says, "the cat bites herself into her tail". But if one can live with that deficiency, it works perfectly.

We first integrate function f3(x) from 1 to 6 and make the shade "lightgrey". Then we integrate f2(x) over the same range.

Finally the integrals (which, in this case are areas) are assigned to variables **lar** (for **l**eft **ar**ea) and **rar** (for **r**ight **ar**ea).



Now one again can "play" with the slider and see not only the graphical effect but also how it looks numerically.



As you can see one can change within the G&G-application the function, the number of maximum possible steps (replace 9 with 19 in the formula for n), the boundaries ...

4. Riemann's Concept of Integration

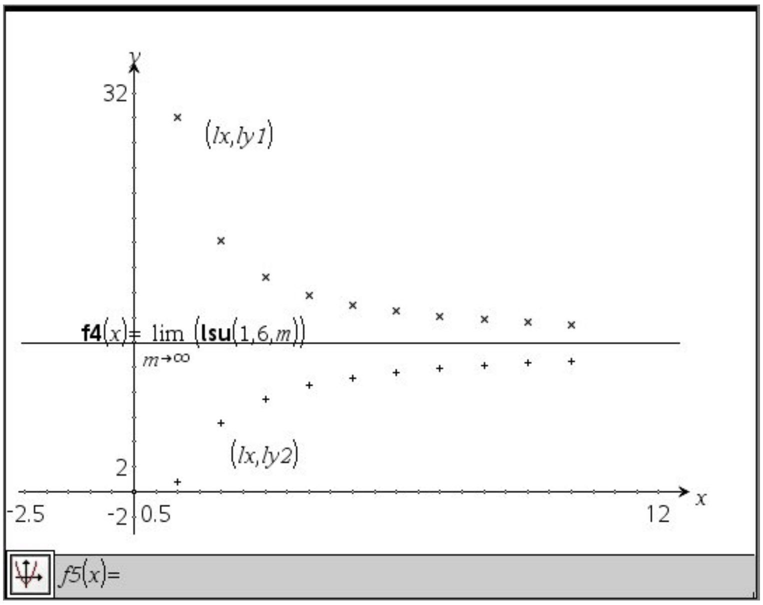
Looking at the areas we see that the **lar**s are decreasing and the **rar**s are increasing when **n** is growing. To make this more intuitive we open a L & S screen. In columns A, B and C we initialize an automated data capture of variables **n**, **lar** and **rar**. Then we return to our G & G screen and pull the slider gently from **n** = 1 to **n** = 10. The three columns are filled with data as we see in the next figure. For plotting reasons we assign list names **lx**, **ly1** and **ly2** to columns A through C. And finally (and that is just for comparison reason) we let the sequence of the **lsu** function (could as well be the **rsu** function) be calculated in column D. And we see that the **lar** values are equal to the values of the **lsu** function. (And so it would be with **rar** and **rsu**.)

(To be honest, the screen to the right does not always result so nicely. Normally the automated data capture catches some triples (**n**, **lar**, **rar**) several times and can also ignore some triples. So one normally has to make several attempts to get all values of **n**. Deleting duplicate triples is not a problem then. If that is all OK one can complete column D.)

	A lx	B ly1	C ly2	D	E
◆	=capture(n,1	=capture(lar	=capture(rar	=seq(lsu(1,6,k),k,1.,10)	
1	1	30	.833333	30.	
2	2	20.1042	5.52083	20.1042	
3	3	17.1914	7.46914	17.1914	
4	4	15.8073	8.51563	15.8073	
5	5	15.	9.16667	15.	
6	6	14.4715	9.61034	14.4715	
7	7	14.0986	9.93197	14.0986	
8	8	13.8216	10.1758	13.8216	
9	9	13.6077	10.3669	13.6077	
10	10	13.4375	10.5208	13.4375	
11					
12					

B | **ly1**:=*capture*(**lar**,1)

And now finally we open another G & G screen to make scatter plots of the lists **ly1** against **lx** and **ly2** against **lx**. Function f4(x) is defined as limit(**lsu**(1,6,m),m,∞).
We can see that the sequence of the **lar**s and **rar**s (or the **lsu**s and **rsu**s) seem to converge to one value which is then called the (definite) **integral of f1(x)** between the bounds 1 and 6.



References:

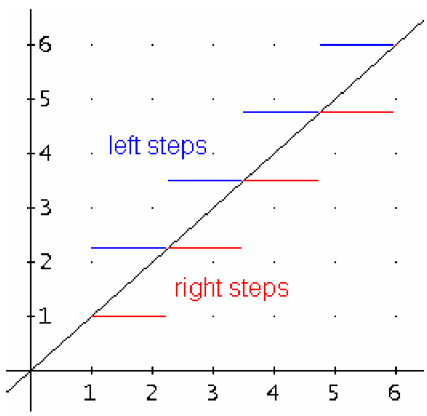
[1] Josef Böhm & Wolfgang Pröpper, Einführung des Integralbegriffs mit dem TI-92, bk teachware 1999
[2] J. Böhm, W. Pröpper, Exploring Integration with the TI-89/92/92+, bk teachware 2000

Remarks:

1. Please have mercy with my English as I am not a native English speaker.
2. The screen shots and the attached TI-Nspire file are based on the "old" Nspire CAS (Build 1.0.500), because I did not get the new version running properly.
3. Do not try to run this example on an "old" handheld. Its performance is much too weak!

Many times in earlier DNLs I was tempted to transfer DERIVE-activities on the screen of a handheld device. This time it is the other way round. Phillipe´s "stepwise" demonstration of the Riemann sums caught my interest – maybe caused by the fact, that Wolfgang and I were very busy presenting various forms of numerical integration and introducing the fundamental theorem (see references above) some time ago. I wondered if the sum-functions would work in Derive and if they were able to apply the slider bar for the shaded areas. Here are my results. Josef.

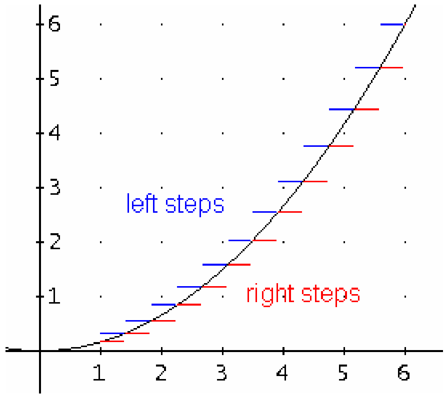
```
       rst(a, b, n, x) :=
         If a ≤ x ≤ b
#1:          (b - a)/n·FLOOR(n·(x - a)/(b - a)) + a
             ?

       lst(a, b, n, x) :=
         If a ≤ x ≤ b
#2:          (b - a)/n·FLOOR(n·(x - a)/(b - a) + 1) + a
             ?

#3:    f1(x) := x

#4:    f1(rst(1, 6, 4))

#5:    f1(lst(1, 6, 4))
```



#6: $f2(x) := \frac{x^2}{6}$

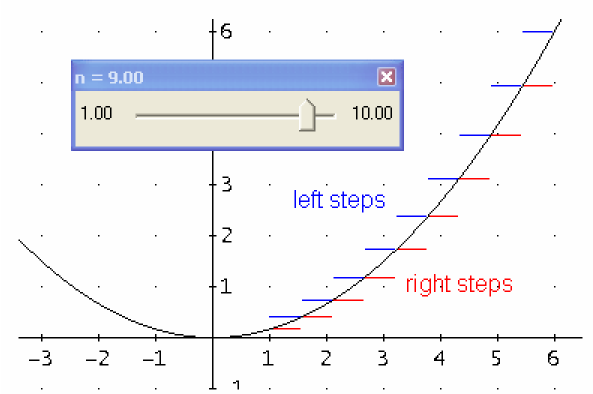
#7: f2(rst(1, 6, 12))

#8: f2(lst(1, 6, 12))



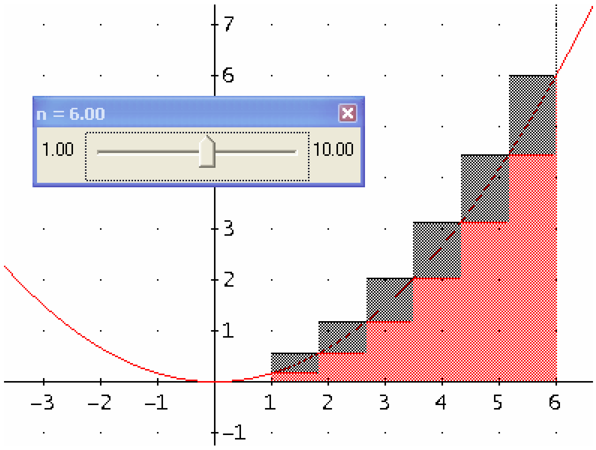
#9: f2(rst(1, 6, n))

#10: f2(lst(1, 6, n))



#11: AreaUnderCurve(f2(lst(1, 6, n)), x, 1, 6, y)

#12: AreaUnderCurve(f2(rst(1, 6, n)), x, 1, 6, y)



#13: $rsu(a, b, m) := \frac{b - a}{m} \cdot \sum_{k=0}^{m-1} f2\left(a + \frac{k \cdot (b - a)}{m}\right)$

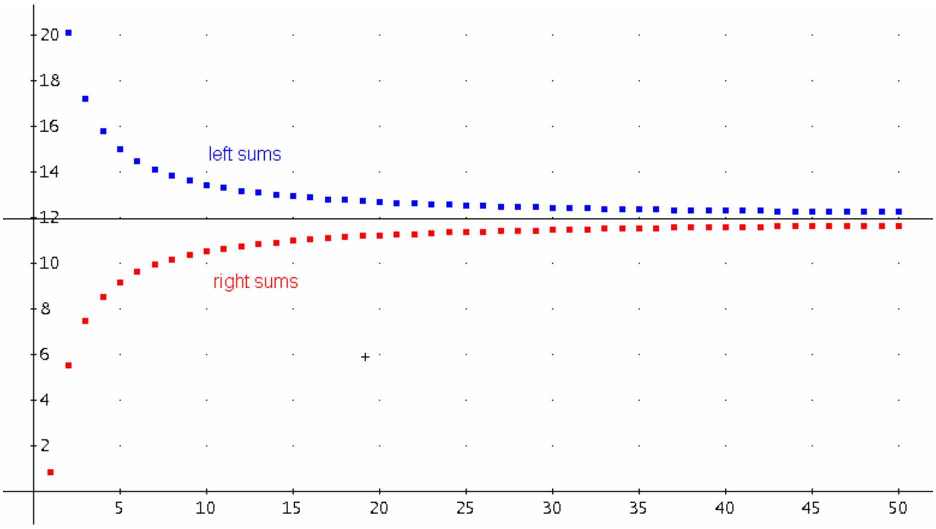
#14: $lsu(a, b, m) := \frac{b - a}{m} \cdot \sum_{k=0}^{m-1} f2\left(a + \frac{(k + 1) \cdot (b - a)}{m}\right)$

#15: sums := TABLE([rsu(1, 6, m), lsu(1, 6, m)], m, 1, 50)

#16: sums↓↓[1, 2]

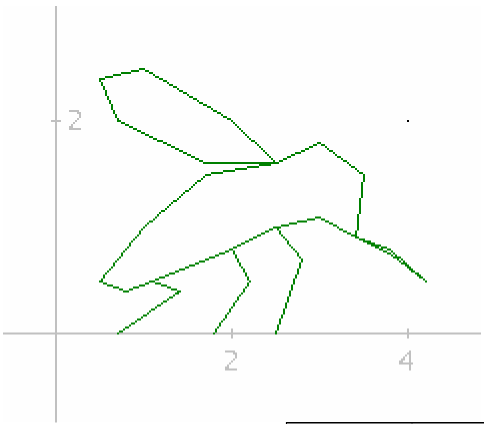
#17: sums↓↓[1, 3]

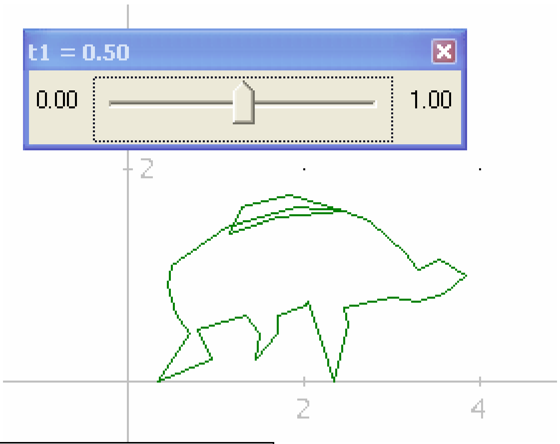
#18: $\left[\lim_{m\to\infty} rsu(1, 6, m), \lim_{m\to\infty} lsu(1, 6, m)\right] = \left[\frac{215}{18}, \frac{215}{18}\right]$

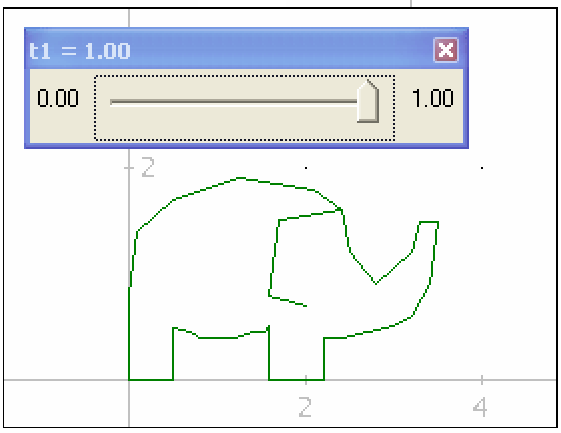


Morphing with DERIVE

I met Maren van Kessel, a colleague of Hubert Weller. She showed a nice "Morphing" following an idea from Hubert: "Make a Parabola from a Straight Line"







I´d like to remind you that we (David Sjöstrand and I) presented in DNL#37, 2000, "A Metamorphosis", where we performed a similar morphing with Excel (and its slider bar) and with the TI-92 (executing a program). In DNL#39 two Lower Austrian students presented their metamorphoses on the TI-92: "Bird to Snail" and "Caterpillar to Butterfly". One of them mentioned in 2000: "Make an Elephant out of a Moscito". The DERIVE slider bar inspires students to morphings on the PC even in 2007.

It is a nice occurrence that in DNL#37 G P Speck contributed an article and he is the author of the next one, too. Josef

Challenger Matrix Problems

G P Speck, Wanganui, New Zealand

So called "*Challenger*" matrix problems appear in various daily newspapers around the world. In the Wanganui Chronicle of 17 June 06 in New Zealand the following such problem authored by Linus Maurer and distributed by King Features Sydicate, Inc. is given in #1 below.

(Note: The format of the problem is modified to meet the matrix format that DERIVE uses but all essential features of the problem type are preserved.)

The object of this problem is to produce, as quickly as possible, any one collection of integers from 1 though 9 for each a, b, …, l such that (see #2):

$$\#1:\quad mm := \begin{bmatrix} 0 & 0 & 0 & 0 & 8 \\ a & 4 & b & c & 7 \\ d & e & 2 & f & 13 \\ 3 & g & h & i & 9 \\ j & k & l & 1 & 9 \\ 9 & 13 & 7 & 9 & 6 \end{bmatrix}$$

$$\#2:\quad [a + 4 + b + c = 7,\ d + e + 2 + f = 13,\ 3 + g + h + i = 9,\ j + k + l + 1 = 9,\ a + d + 3 + j = 9,\ 4 + e + g + k = 13,\ b + 2 + h + l = 7,\ c + f + i + 1 = 9,\ a + e + h + 1 = 6,\ j + g + 2 + c = 8]$$

Define the submatrix m of matrix mm as shown in expression #3:

$$\#3:\quad m := \begin{bmatrix} a & 4 & b & c \\ d & e & 2 & f \\ 3 & g & h & i \\ j & k & l & 1 \end{bmatrix}$$

Thus stated in terms of m and mm, the object of this problem is to produce integers from 1 through 9 for each of the letters a, b, …, l such that the sum of the elements in each row in m is equal to the number to the right of the row in mm; the sum of the elements in each column in m is equal to the number below the column in mm; the sum of the main diagonal elements in m is equal to the number in the lower right of mm; the sum of the off diagonal elements in m is equal to the number in the upper right of mm. Further, in m the four given numbers (in this example 4, 2, 3, 1) must each be selected from the integers 1 through 9 with one of the four in each row, one of the four in each column, and one of the four in each diagonal. Under these requirements there are only eight 4-tuple positions in matrix m where these four integers may be placed:

```
(m(1,1),m(2,3),m(3,4),m(4,2))
(m(1,2),m(2,3),m(3,1),m(4,4))
(m(1,3),m(2,1),m(3,2),m(4,4))

(m(1,4),m(2,1),m(3,3),m(4,2))
(m(1,3),m(2,2),m(3,4),m(4,1))
(m(1,4),m(2,2),m(3,1),m(4,3))

(m(1,1),m(2,4),m(3,2),m(4,3))
(m(1,2),m(2,4),m(3,3),m(4,1))
```

An Obvious Observation: It is a trivial matter to construct any number of *Challenger* matrix problems each of which has at least one matrix solution.

Now our objective is a much more ambitious one that stated in the text preceding #2 above in that we wish to construct a DERIVE program which will give **ALL** solutions in the integers 1 through 9 for any *Challenger* matrix problem. Our DERIVE program will consist first of “solving” a system of 10 equations in 12 unknowns as given in #2 above, for any complex numbers of the form x + yî. This will be done by solving for each of the letters (a through l) in terms of either i, k & l or h, k & l. Fortunately, DERIVE handles this problem easily so that we do not have to resort to Cramer´s rule after consideration of dependence and other issues or tedious soul-destroying reduction methods. After this is done, however, we then can utilize DERIVE´s brute force power to produce a frontal attack to bypass any tedious or clever Diophantine analysis to select out those solutions where each of (a through l) is one of the integers 1 through 9. It is in fact the case that DERIVE needs consider only 729 triples of integers (i,k,l) or (h,k,l) in yielding all of the appropriate integer solutions to a given *Challenger* matrix problem.

Returning now to #1 and #2 above for the purpose of illustration, we solve the 10 equations in 12 unknowns from #2 writing the SOLVE-statement in #4 below and then performing Simplify/Basic on it to get #5 below:

```
#4:   SOLVE([a + 4 + b + c = 7, d + e + 2 + f = 13, 3 + g + h + i = 9, j + k + l
        + 1 = 9, a + d + 3 + j = 9, 4 + e + g + k = 13, b + 2 + h + l = 7, c + f
        + i + 1 = 9, a + e + h + 1 = 6, j + g + 2 + c = 8], [a, b, c, d, e, f,
        g, h, j])

#5:   [a = -i - k + 6 ∧ b = -k - l + 7 ∧ c = i + 2·k + l - 10 ∧ d = i + 2·k + l
        - 8 ∧ e = i + 1 ∧ f = - 2·i - 2·k - l + 18 ∧ g = -i - k + 8 ∧ h = k - 2
        ∧ j = -k - l + 8]
```

Now after placing #5 in the Author line, modify it by replacing each “^” with a “,” and each “=” with a “:=” to get #6 below:

Note: alternatively, we could double click on line #5, perform the modifications indicated above in the Vector Setup appearing, and Enter to replace line #5 with the result in line #6 below.

```
#6:   [a := -i - k + 6, b := -k - l + 7, c := i + 2·k + l - 10, d := i + 2·k + l -
        8, e := i + 1, f := - 2·i - 2·k - l + 18, g := -i - k + 8, h := k - 2, j :=
        -k - l + 8]
```

Next Simplify/Basic #1 above to get:

$$\#7:\quad \begin{bmatrix} 0 & 0 & 0 & 0 & 8 \\ -i - k + 6 & 4 & -k - l + 7 & i + 2\cdot k + l - 10 & 7 \\ i + 2\cdot k + l - 8 & i + 1 & 2 & -2\cdot i - 2\cdot k - l + 18 & 13 \\ 3 & -i - k + 8 & k - 2 & i & 9 \\ -k - l + 8 & k & l & 1 & 9 \\ 9 & 13 & 7 & 9 & 6 \end{bmatrix}$$

The above #7 is indeed interesting; for any i, k, l whether they be integers, or rational numbers, or irrational numbers, or complex numbers – the sum of the first four numbers in row 2 equals the number 7 on the right, and similarly for the first four numbers in rows 3, 4 and 5. Similar statements can also be made for the sums of the columns and the sums of the two diagonals.

Now for the Diophantine analysis – we must select each of the i, k & l in #7 as integers from the set 1 through 9 and determine which selections make **EACH** of the integers in the submatrix `m` of `mm` belong to the set of 1 thru 9. It is easy to see that there are precisely 9×9×9 = 729 triples to consider initially as candidates for solutions to a given *Challenger* matrix problem. If each coordinate of an arbitrary triple from the 729 candidates is given in (xx, yy, zz), then we could display the 729 xx-coordinates, the 729 yy-coordinates, and the 729 zz-coordinates in three lines. However, it is easy to make use of the periodic nature of xx, yy, and zz coordinates to create functions which we can use rather than actually listing all 729 triples (xx,yy,zz), and the author has done this in the following lines.

`FloorType(x)` is just like `FLOOR(x)` except the endpoint is on the right side of each line segment on its graph.

```
#8:   FloorType(x) := -1 - FLOOR(-x)
```

The following `fxx(x)` gives xx matrix entries:

```
      fxx(x) :=
        If x < 1 ∨ x > 729
#9:        0
           If 1 ≤ x ≤ 729
              - FLOOR(- x/81)
```

The following `FloorType2(x)` gives the beginning of yy matrix entries:

```
      FloorType2(x) :=
#10:    If 1 ≤ x ≤ 81
           - FLOOR(- x/9)
```

The following `fyy(x)` gives the completion of the yy matrix entries:

```
      FloorType3(x) :=
        If x < 1 ∨ x > 81
#11:       0
           FloorType2(x)
```

$$\#12:\quad \text{fyy}(x) := \text{FloorType3}\left(x - 81\cdot\text{FloorType}\left(\frac{x}{81}\right)\right)$$

```
      fzz1(x) :=
        If 0 ≤ x ≤ 9
#13:       x
           0
```

$$\#14:\quad \text{fzz}(x) := \text{fzz1}\left(x - 9\cdot\text{FloorType}\left(\frac{x}{9}\right)\right)$$

$$\#15:\quad \text{vec} := \begin{bmatrix} \text{VECTOR(fxx(x), x, 1, 729)} \\ \text{VECTOR(fyy(x), x, 1, 729)} \\ \text{VECTOR(fzz(x), x, 1, 729)} \end{bmatrix}$$

Note: We could place `vec=` in the Author line and Enter to display all 729 (xx,yy,zz) triples.

The definitions in the following lines enable us to find the number of solutions and all of the solutions for any given *Challenger* matrix problem.

#16: $aa := \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$

#17: $bb := \begin{bmatrix} 9 & 9 & 9 & 9 \\ 9 & 9 & 9 & 9 \\ 9 & 9 & 9 & 9 \\ 9 & 9 & 9 & 9 \end{bmatrix}$

```
#18:  m1(x, y, z) :=

#19:  n1(x, y, z) := VECTOR(VECTOR((m1(x, y, z))    , v, 1, 4), u, 2, 5)
                                                u,v

#20:  b_b := q := q + 1

#21:  a_a := aa ≤ n1(fxx(r), fyy(r), fzz(r)) ≤ bb

      NumSol(p) :=
        Prog
          q := 1
          r := p
          Loop
#22:        If a_a ∧ n1(fxx(r), fyy(r), fzz(r)) = FLOOR(n1(fxx(r), fyy(r), fzz(r)))
               q := q + 1
               If r ≥ 729
                  RETURN ["Number of Solutions = "; q - 1]
            r := r + 1

      AllSol(VecDim) :=
        Prog
          tt := VECTOR(φ, x, 1, VecDim)
          q := 1
          r := 1
#23:      Loop
            If a_a ∧ n1(fxx(r), fyy(r), fzz(r)) = FLOOR(n1(fxx(r), fyy(r), fzz(r)))
               [tt↓q := [q, r, [fxx(r), fyy(r), fzz(r)], m1(fxx(r), fyy(r), fzz(r))], b_b]
               If r ≥ 729
                  RETURN tt
            r := r + 1
```

We now return to the particular *Challenger* matrix problem given in #1 above and resume, using the result in #7 above, to produce **ALL** solutions in integers 1 thru 9 for this *Challenger* problem.

#24: $m1(i, k, l) := \begin{bmatrix} 0 & 0 & 0 & 0 & 8 \\ -i - k + 6 & 4 & -k - l + 7 & i + 2\cdot k + l - 10 & 7 \\ i + 2\cdot k + l - 8 & i + 1 & 2 & -2\cdot i - 2\cdot k - l + 18 & 13 \\ 3 & -i - k + 8 & k - 2 & i & 9 \\ -k - l + 8 & k & l & 1 & 9 \\ 9 & 13 & 7 & 9 & 6 \end{bmatrix}$

The number of solutions for our particular *Challenger* problem is given in the following line placing `NumSol(1)` = in the Author line and pressing Enter.

$$\#25: \quad \text{NumSol}(1) = \begin{bmatrix} \text{Number of Solutions =} \\ 2 \end{bmatrix}$$

All solutions, in this case 2, for our particular problem are given in the following line placing `AllSol(2)` = in the Author line and pressing Enter.

$$\#26: \quad \text{AllSol}(2) = \begin{bmatrix} 1 & 29 & [1,\ 4,\ 2] & \begin{bmatrix} 0 & 0 & 0 & 0 & 8 \\ 1 & 4 & 1 & 1 & 7 \\ 3 & 2 & 2 & 6 & 13 \\ 3 & 3 & 2 & 1 & 9 \\ 2 & 4 & 2 & 1 & 9 \\ 9 & 13 & 7 & 9 & 6 \end{bmatrix} \\ 2 & 102 & [2,\ 3,\ 3] & \begin{bmatrix} 0 & 0 & 0 & 0 & 8 \\ 1 & 4 & 1 & 1 & 7 \\ 3 & 3 & 2 & 5 & 13 \\ 3 & 3 & 1 & 2 & 9 \\ 2 & 3 & 3 & 1 & 9 \\ 9 & 13 & 7 & 9 & 6 \end{bmatrix} \end{bmatrix}$$

****

At this point, with enough background information given to define the *Challenger* problem type and to see the direction we have taken above in solving a particular such problem, we give below a DERIVE program recipe which with all the 26 lines above, will enable us to solve ANY *Challenger* matrix problem; but first we give the following definitions useful in entering and solving these problems.

$$\#27: \quad \begin{bmatrix} f13 := \begin{bmatrix} 0 & 0 & 0 & 0 & \phi \\ \phi & a & b & c & \phi \\ d & e & \phi & f & \phi \\ g & h & i & \phi & \phi \\ j & \phi & k & l & \phi \\ \phi & \phi & \phi & \phi & \phi \end{bmatrix}, & f23 := \begin{bmatrix} 0 & 0 & 0 & 0 & \phi \\ a & \phi & b & c & \phi \\ d & e & \phi & f & \phi \\ \phi & g & h & i & \phi \\ j & k & l & \phi & \phi \\ \phi & \phi & \phi & \phi & \phi \end{bmatrix}, & f31 := \begin{bmatrix} 0 & 0 & 0 & 0 & \phi \\ a & b & \phi & c & \phi \\ \phi & d & e & f & \phi \\ g & \phi & h & i & \phi \\ j & k & l & \phi & \phi \\ \phi & \phi & \phi & \phi & \phi \end{bmatrix} \end{bmatrix}$$

$$\#28: \quad \begin{bmatrix} f41 := \begin{bmatrix} 0 & 0 & 0 & 0 & \phi \\ a & b & c & \phi & \phi \\ \phi & d & e & f & \phi \\ g & h & \phi & i & \phi \\ j & \phi & k & l & \phi \\ \phi & \phi & \phi & \phi & \phi \end{bmatrix}, & f32 := \begin{bmatrix} 0 & 0 & 0 & 0 & \phi \\ a & b & \phi & c & \phi \\ d & \phi & e & f & \phi \\ g & h & i & \phi & \phi \\ \phi & j & k & l & \phi \\ \phi & \phi & \phi & \phi & \phi \end{bmatrix}, & f42 := \begin{bmatrix} 0 & 0 & 0 & 0 & \phi \\ a & b & c & \phi & \phi \\ d & \phi & e & f & \phi \\ \phi & g & h & i & \phi \\ j & k & \phi & l & \phi \\ \phi & \phi & \phi & \phi & \phi \end{bmatrix} \end{bmatrix}$$

#29: $\left[f14 := \begin{bmatrix} 0 & 0 & 0 & 0 & \phi \\ \phi & a & b & c & \phi \\ d & e & f & \phi & \phi \\ g & \phi & h & i & \phi \\ j & k & \phi & l & \phi \\ \phi & \phi & \phi & \phi & \phi \end{bmatrix}, f24 := \begin{bmatrix} 0 & 0 & 0 & 0 & \phi \\ a & \phi & b & c & \phi \\ d & e & f & \phi & \phi \\ g & h & \phi & i & \phi \\ \phi & j & k & l & \phi \\ \phi & \phi & \phi & \phi & \phi \end{bmatrix} \right]$

#30: $v1 := \mathrm{VECTOR}(mm_{p,1} + mm_{p,2} + mm_{p,3} + mm_{p,4} = mm_{p,5}, p, 2, 5)$

#31: $v2 := \mathrm{VECTOR}(mm_{2,p} + mm_{3,p} + mm_{4,p} + mm_{5,p} = mm_{6,p}, p, 1, 4)$

#32: $v3 := [mm_{2,1} + mm_{3,2} + mm_{4,3} + mm_{5,4} = mm_{6,5}]$

#33: $v4 := [mm_{5,1} + mm_{4,2} + mm_{3,3} + mm_{2,4} = mm_{1,5}]$

Now the aforementioned recipe for solving ANY *Challenger* problem consists of placing in the Author line in sequence the symbol `si=` (i=1,…,7), unless otherwise instructed, for each of the `s1` thru `s7` given below, with Enter being pressed after each `si=` is placed in the Author line.

#34: s1 :=

(1) The recipe for solving ANY Challenger problem consists of placing in the Author line in sequence the symbol si= , unless otherwise instructed, for each of s1 through s7 with Enter being pressed after each entry of si=.

(2) You have placed the symbol s1= in the Author line and Entered it to execute the following line on the letters a through l:

[a :=, b :=, c :=, d :=, e :=, f :=, g :=, h :=, i :=, j :=, k :=, l :=]

[a :=, b :=, c :=, d :=, e :=, f :=, g :=, h :=, i :=, j :=, k :=, l :=]

(3) To duplicate a given newspaper Challenger matrix problem into a Derive matrix where the newspaper matrix has one of its 4 given number entries in the (2,p) position and a second of its 4 given numbers in the (3,q) position, place mm:=fpq in the Author line and press Enter. Next place mm= in the Author line and press Enter. Then double click the resulting mm=[6x5 matrix] and in the Author line, replace mm= with mm:= , and replace each φ appearing with its corresponding number from the newspaper Challenger problem. Finally, press Enter to display the replacement mm:=[new newspaper 6x5 matrix].

#35: s2 :=

You have placed the symbol s2= in the Author line and Entered it

to determine the vector w0 and number c_c needed subsequently.

w0 := [a, b, c, d, e, f, g, h, j]

c_c := 1

IF($mm_{2,1} > 0 \wedge mm_{3,4} > 0$, (w0 := [a, b, c, d, e, f, g, i, j]) ∧ c_c := 2)

IF($mm_{2,2} > 0 \wedge mm_{3,4} > 0$, (w0 := [a, b, c, d, e, f, g, i, j]) ∧ c_c := 2)

[[w0= / w0], [c_c= / c_c]]

#36: s3 :=

(1) Place the symbol s4 in the Author line and then click = .

(2) Next double click on the result of s4 and replace it after modifying it by replacing each ∧ with , and each = with := and then pressing Enter.

(3) Finally, place the symbol s5 in the Author line and then click =.

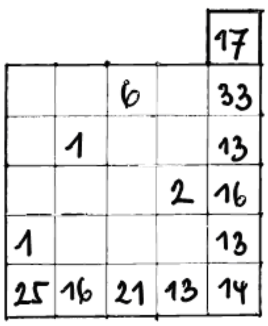
#37: s4 := SOLVE($[v1_1, v1_2, v1_3, v1_4, v2_1, v2_2, v2_3, v2_4, v3_1, v4_1]$, w0)

#38: s5 := IF(c_c = 1, m1(i, k, l) := mm, m1(h, k, l) := mm)

#39: s6 := NumSol(1)

#40: s7 := [All Solutions =, AllSol(q - 1)]

An application of the recipe above is given below for solving the *Challenger* problem which is represented by the matrix on the right:



$$\begin{bmatrix} 0 & 0 & 0 & 0 & 17 \\ a & b & 6 & c & 33 \\ d & 1 & e & f & 13 \\ g & h & i & 2 & 16 \\ 1 & j & k & l & 13 \\ 25 & 16 & 21 & 13 & 14 \end{bmatrix}$$

We follow the instructions `s1` thru `s7` and start with simplifying `s1=` . We can read how to proceed (I don´t copy the screen) and do according to intruction (3) given in `s1`:

#43: mm := f32

`mm=` delivers a "palette" for editing our particular problem and defining matrix `mm` (# 45).

We execute `s2=` and then `s3=` followed by `s4=`.

We do again some editing in order to obtain the several assignments in #49.

`s5=` returns the matrix with only three variables.

Finally `s6=` and `s7=` present the solution(s) of the *Challenger* problem.

$$\#44:\quad mm = \begin{bmatrix} 0 & 0 & 0 & 0 & \phi \\ a & b & \phi & c & \phi \\ d & \phi & e & f & \phi \\ g & h & i & \phi & \phi \\ \phi & j & k & l & \phi \\ \phi & \phi & \phi & \phi & \phi \end{bmatrix}$$

$$\#45:\quad mm := \begin{bmatrix} 0 & 0 & 0 & 0 & 17 \\ a & b & 6 & c & 33 \\ d & 1 & e & f & 13 \\ g & h & i & 2 & 16 \\ 1 & j & k & l & 13 \\ 25 & 16 & 21 & 13 & 14 \end{bmatrix}$$

$$\#48:\quad s4 = [a = -i - l + 13 \wedge b = l + 8 \wedge c = i + 6 \wedge d = 2\cdot i + k + l - 8 \wedge e = -i - k + 15 \wedge f = -i - l + 5 \wedge g = -i - k + 19 \wedge h = k - 5 \wedge j = -k - l + 12]$$

$$\#49:\quad [a := -i - l + 13,\ b := l + 8,\ c := i + 6,\ d := 2\cdot i + k + l - 8,\ e := -i - k + 15,\ f := -i - l + 5,\ g := -i - k + 19,\ h := k - 5,\ j := -k - l + 12]$$

$$\#50:\quad s5 = m1(i, k, l) := \begin{bmatrix} 0 & 0 & 0 & 0 & 17 \\ -i - l + 13 & l + 8 & 6 & i + 6 & 33 \\ 2\cdot i + k + l - 8 & 1 & -i - k + 15 & -i - l + 5 & 13 \\ -i - k + 19 & k - 5 & i & 2 & 16 \\ 1 & -k - l + 12 & k & l & 13 \\ 25 & 16 & 21 & 13 & 14 \end{bmatrix}$$

$$\#51:\quad s6 = \begin{bmatrix} \text{Number of Solutions =} \\ 3 \end{bmatrix}$$

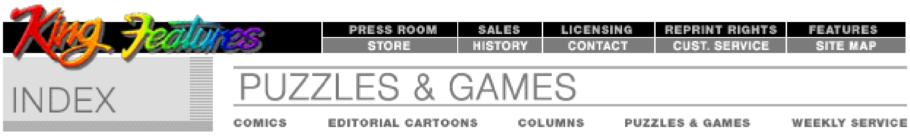
$$\#52:\quad s7 = \left[\ \text{All Solutions=,}\ \begin{bmatrix} 1 & 217 & [3, 7, 1] & \begin{bmatrix} 0 & 0 & 0 & 0 & 17 \\ 9 & 9 & 6 & 9 & 33 \\ 6 & 1 & 5 & 1 & 13 \\ 9 & 2 & 3 & 2 & 16 \\ 1 & 4 & 7 & 1 & 13 \\ 25 & 16 & 21 & 13 & 14 \end{bmatrix} \\ 2 & 226 & [3, 8, 1] & \begin{bmatrix} 0 & 0 & 0 & 0 & 17 \\ 9 & 9 & 6 & 9 & 33 \\ 7 & 1 & 4 & 1 & 13 \\ 8 & 3 & 3 & 2 & 16 \\ 1 & 3 & 8 & 1 & 13 \\ 25 & 16 & 21 & 13 & 14 \end{bmatrix} \\ 3 & 235 & [3, 9, 1] & \begin{bmatrix} 0 & 0 & 0 & 0 & 17 \\ 9 & 9 & 6 & 9 & 33 \\ 8 & 1 & 3 & 1 & 13 \\ 7 & 4 & 3 & 2 & 16 \\ 1 & 2 & 9 & 1 & 13 \\ 25 & 16 & 21 & 13 & 14 \end{bmatrix} \end{bmatrix}\ \right]$$

The next *Challenger* problem (which has 249 solutions!) is given for solution by the reader using the RECIPE given above.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 20 \\ a & 5 & b & c & 20 \\ d & e & 5 & f & 20 \\ 5 & g & h & i & 20 \\ j & k & l & 5 & 20 \\ 20 & 20 & 20 & 20 & 20 \end{bmatrix}$$

Searching the web for *Challenger* problems I found the website of King features `www. Kingfeatures.com/index.htm` and a Demo *Challenger*.

Try to solve it!



challenger® THE INTERNATIONAL CROSSNUMBER GAME

DIRECTIONS:
Fill each square with a number, one through nine.
- Horizontal squares should add to totals on right.
- Vertical squares should add to totals on bottom.
- Diagonal squares through center should add to total in upper and lower right.

THERE MAY BE MORE THAN ONE SOLUTION.

Today's Challenge
Time 0 Minutes
49 Seconds
Your Working
Time Minutes
Seconds

				9
		4		11
1				5
	4			23
			3	9
17	10	15	6	18

9-6

G P Speck is from Wanganui. Some years ago my wife and I had the occasion to visit New Zealand and we had a boat trip on the Wanganui River and travelled along it to the City of Wanganui. There is a wonderful museum containing a rich Maori Art collection. The pictures show the Wanganui River, pittoresque village Jerusalem on the River and the famous Bridge to Nowhere. Josef

Johann Wiesenbauer, Vienna

Hi,

There seems to be some kind of misunderstanding. When discussing general powers A^n here , we are not talking about general matrices A , but about general exponents n.

No mathematician (and no CAS for that matter) will ever be able to compute A^n and exp(A) for an arbitrary given matrix A and for a general n (exactly, mind you!), for the simple reason that we usually cannot determine the exact roots of the characteristic polynomial of A if its degree is greater than 4. The computation of those roots, also called eigenvalues of A, is inevitable though when computing A^n for an unspecified nonnegative integer n.

Cheers,
Johann

R. Schorn's challenge (from DNL #13, p.3) revisited

Aleksey Tetyorko

Hi!

Better later then never - try my versions of MAXTERM function. I've run it in 3.11, 4.11, and 5.06 versions of Derive. Annotations give the running times - my computer is old (Pentium, WIN95, 16M RAM etc).

But the function is not so bad. Sorry, I do not know, how I could it compute 13 years ago.

$$\text{MAXTERM5\_AUX}(u,\ x,\ m\_,\ n\_,\ c\_) := \text{ITERATE}\left(\text{IF}\left(n\_ < \text{DIMENSION}(u),\ \text{IF}\left(m\_{_2} > m\_{_1},\ \left[\left[m\_{_2},\ \lim_{x\to 1} u_{n\_+1}\right],\ n\_ + 1,\ u_{n\_}\right],\ \left[\left[m\_{_1},\ \lim_{x\to 1} u_{n\_+1}\right],\ n\_ + 1,\ c\_\right]\right),\ \text{IF}\left(m\_{_2} > m\_{_1},\ \left[m\_{_2},\ n\_,\ u_{n\_}\right],\ \left[m\_{_1},\ n\_,\ c\_\right]\right)\right),\ [m\_,\ n\_,\ c\_],\ \left[\left[\lim_{x\to 1} u_1,\ \lim_{x\to 1} u_2\right],\ 2,\ u_1\right],\ \text{DIMENSION}(u) - 1\right)$$

$$\text{MAXTERM5}(u,\ n,\ x) := \text{MAXTERM5\_AUX}(\text{TERMS}(\text{EXPAND}(u^n),\ x),\ x)$$

$$\text{MAXTERM5}(4\cdot x^3 + 3\cdot x^2 + 2\cdot x + 1,\ 100,\ x)$$

$$[398243667203410623024848236684603963137619260356309333266022065462985155939604195463780443664821745,\ 301,\ 398243667203410623024848236684603963137619260356309333266022065462985155939604195463780443664821745\cdot x^{200}]$$

Simplification time 25.0 sec with v. 3.11
Simplification time 21.5 sec. with v. 4.11
Simplification time 37.0 sec with v. 5.06

Simplification time 2.23 sec with v. 6.10

$$\text{MAXTERM5}(x + 1,\ 2,\ x) = [2,\ 3,\ 2\cdot x]$$

Aleksey

Johann Wiesenbauer

Hello Aleksey,
Okay, you have proven that it is possible to get the desired result within a few seconds in those ancient versions of Derive. I'm not sure though if there are people who will find this piece of information useful. As for me, I have to confess that I'm overglad that I'm no longer forced to deal the unreadable programs of that time. In fact, there is even one program of mine (the function `parts(n)` in `CombinatorialFunctions.mth`) which I have never translated into the new programming code of Derive 5 (or more recent), because I simply couldn't decypher it anymore.

Anyway, if you want to know how Richard Schorn computed the coefficient at issue then (without the corresponding power of x), you could look it up here

http://www.austromath.at/dug/dnl01.php

in the DNL quoted in subject (though on page 12).

Cheers,
Johann

Jim FitzSimons

Johann, your old `parts` still works and is a lot faster than my `parts`.
My `parts` is included in `CommbinatorialFunctions.mth`.

```
MAXTERMS(x + 1, 2, x) = [2, 3, 2·x]

PARTS_AUX(n, m) :=
  If n < 2·m
     1
     1 + Σ(PARTS_AUX(n - k_, k_), k_, m, FLOOR(n, 2))

PARTSJ(n) :=
  If n < 0
     0
     PARTS_AUX(n, 1)

PARTSJ(50) = 204226

needs 7.72 sec

PARTS(50)

204226
```

Your `parts` is so fast, it can not be measured (0.016 sec.).
Jim,

Johann Wiesenbauer

Jim,
Thank you very much for your kind comment, but the credit for the outstanding performance of my `parts(n)` goes mostly to Ramanujan and Rademacher, who found the underlying formula. I took me some days then just to understand how it works, but I have never understood how someone can come up with such an amazing formula.

Cheers,
Johann

Aleksey Tetyorko

Hi!

Mutually recursive functions and Lisp are my first love and my hobby, and I treat the Schorn's problem as the programmer's puzzle.

The MAXTERM5 function can be rewritten in imperative (while-loop) manner, but for what?! Derive as one of Lisp (functional) programming systems has the tail recursion elimination etc. But... the "while-loop" function follows.

```
      MAXTERM5_AUX_L(u, x, m_, n_, c_, n__, n_max) :=
        Prog
          m_ := [LIM(u↓1, x, 1), LIM(u↓2, x, 1)]
          n_ := 2
          c_ := u↓1
          n__ := 0
          n_max := DIMENSION(u) - 1
          Loop
            If n__ ≥ n_max exit
            If n_ < DIMENSION(u)
               If m_↓2 > m_↓1
                  Prog
#1:                 m_ := [m_↓2, LIM(u↓(n_ + 1), x, 1)]
                    c_ := u↓n_
                    n_ :+ 1
                  Prog
                    m_ := [m_↓1, LIM(u↓(n_ + 1), x, 1)]
                    n_ :+ 1
               If m_↓2 > m_↓1
                  Prog
                    m_ := m_↓2
                    c_ := u↓n_
                  m_ := m_↓1
            n__ :+ 1
          [m_, n_, c_]

                                                            n
#2:   MAXTERM5L(u, n, x) := MAXTERM5_AUX_L(TERMS(EXPAND(u )), x), x)

#3:   MAXTERM5L(x + 1, 2, x) = [2, 3, 2·x]

                    3      2
#4:   MAXTERM5L(4·x  + 3·x  + 2·x + 1, 100, x)

#5:   [398243667203410623024848236684603963137619260356309333266022065462985155939604195463780443664821745, 301,
                                                                                                              200
        398243667203410623024848236684603963137619260356309333266022065462985155939604195463780443664821745·x    ]
```

This version was obtained from the recursive one by the straightforward expansion of ITERATE using the LOOP construction. One can improve the readability by some rearranging.

Aleksey

PS.

```
iter(u, x, x0, n, x_, n_) :=
  Prog
    x_ := x0
    n_ := 0
    Loop
      If n_ ≥ n exit
      x_ := LIM(u, x, x_)
      n_ :+ 1
    x_

iter(x + 1, x, 0, 20) = 20
```

```
iters(u, x, x0, n, x_, n_, r_) :=
  Prog
    x_ := x0
    r_ := [x_]
    n_ := 0
    Loop
      If n_ ≥ n exit
      x_ := LIM(u, x, x_)
      r_ := APPEND(r_, [x_])
      n_ :+ 1
    r_

iters(x + 1, x, 0, 20)

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]
```

Strange Derivatives on the TI-Device??

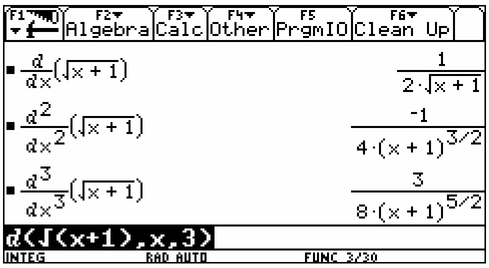
Hubert Langlotz

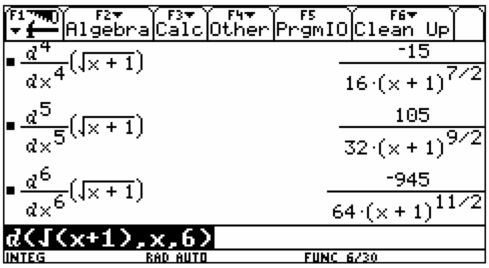
Hello Josef, hello Rainer (Geyer),

one of my colleagues came across the following problem and I couldn´t give any support. In my opinion it should be a bug. Do you have any advice?

Hubert

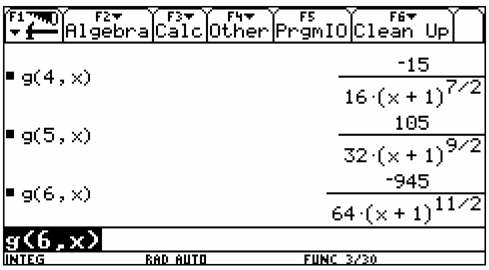
The problem is: Find the n-th derivative of $f(x)=\sqrt{x+1}$.





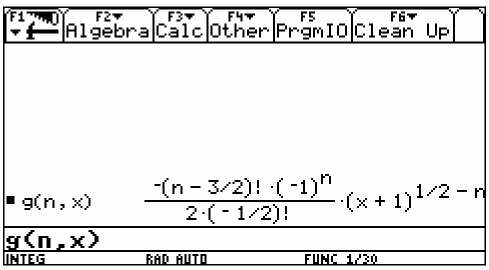
The conjecture is: $f^{n}(x)=(-1)^{n-1}\frac{\prod_{i=1}^{n-1}(2i-1)}{2^{n}}(x+1)^{-\frac{2n-1}{2}}$.

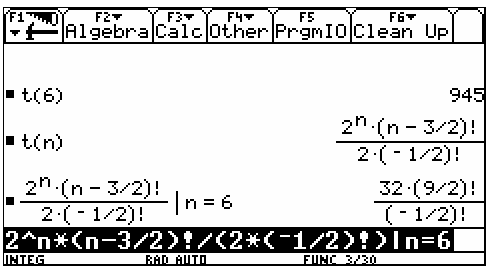
Having stored this expression as g(n,x), one obtains the following correct results:



`g(n,x)` gives a surprising output.

Saving the product only as `t(n)` one faces the same problem:





This is my attempt of an explanation, Josef.

Dear Hubert (and Rainer and colleague, of course),

Derive delivers the following:

$$\#1: \quad \prod_{i=1}^{n-1} (2\cdot i - 1) = \frac{2^{n-1}\cdot\left(n - \frac{3}{2}\right)!}{\sqrt{\pi}}$$

$$\#2: \quad \frac{2^{6-1}\cdot\left(6 - \frac{3}{2}\right)}{\sqrt{\pi}} = 945$$

This is a „beautiful“ example where we need proceeding manually because „the calculator – or the implemented program – is failing“.

We prove the conjecture by induction:

1. step: $$f^{(n)}(x) = (-1)^{n-1}\frac{\prod_{i=1}^{n-1}(2i-1)}{2^n}(x+1)^{-\frac{2n-1}{2}}$$

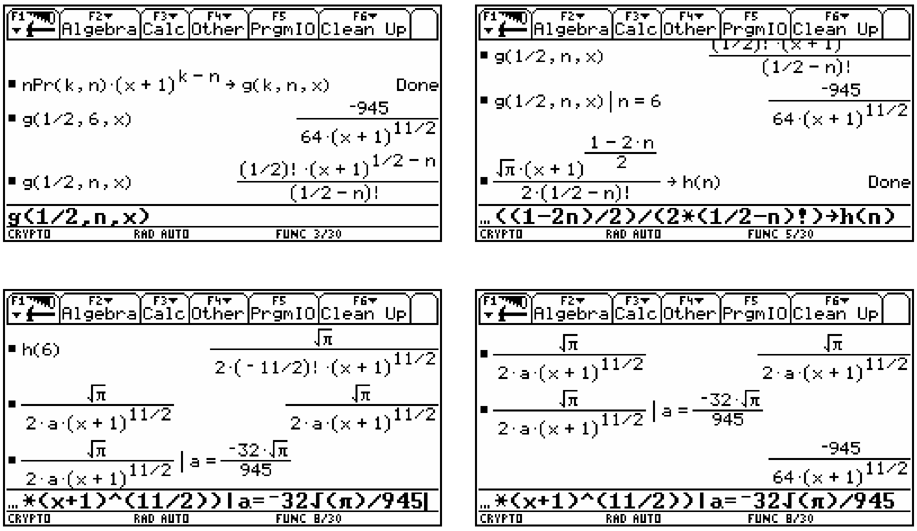
2. step: $$f^{(n+1)}(x) = (-1)^{n}\frac{\prod_{i=1}^{n}(2i-1)}{2^{n+1}}(x+1)^{-\frac{2(n+1)-1}{2}}$$

3. step: $$(f^{(n)}(x))' = (-1)^{n-1}\frac{\prod_{i=1}^{n-1}(2i-1)}{2^n}\cdot\left(-\frac{2n-1}{2}\right)\cdot(x+1)^{-\frac{2n-1}{2}-1} = f^{(n+1)}(x)$$

Answer: The general formula for the n-the derivative of x^k is given by

$$\frac{d^n}{dx^n}x^k = (x^k)^{(n)} = \binom{k}{n} n!\cdot x^{k-n}$$

I transmit this formula on the V200 and calculate the 6th derivative. As the “inner derivative” is 1, we have no conflicts with the chain rule replacing x with x+1.



The reason for the "bug" – which is no bug at all – is the fact that the V200 and with it the recent version of the NSpire are not as "clever" as Derive and they both don´t know how to handle faculties of fractions. There is the GAMMA-function in the background and with it the knowledge that $(1/2)! = \sqrt{\pi}/2$. This should be implemented in further versions of the handheld TIs.

#1: $g(k, n, x) := \mathrm{PERM}(k, n)\cdot(x + 1)^{k - n}$

#2: $g\left(\frac{1}{2}, 6\right) = - \frac{945}{64\cdot(x + 1)^{11/2}}$

#3: $g\left(\frac{1}{2}, n\right) = \frac{\sqrt{\pi}\cdot(x + 1)^{(1 - 2\cdot n)/2}}{2\cdot\left(\frac{1}{2} - n\right)!}$

#4: $\left(- \frac{11}{2}\right)! = - \frac{32\cdot\sqrt{\pi}}{945}$

#5: $\left(\frac{1}{2}\right)! = \frac{\sqrt{\pi}}{2}$

#6: $\left(- \frac{1}{2}\right)! = \sqrt{\pi}$

If we import the knowledge acquired by support of DERIVE into the handheld devices we can immediately see the equivalence of the results.

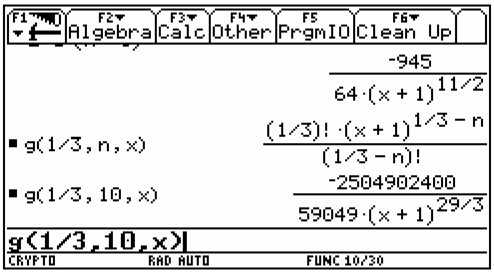
By the way it is very instructive to perform the calculation of Hubert´s product in Derive - STEPWISE.

#6: $g\left(\frac{1}{3}, n\right) = \frac{(x + 1)^{(1 - 3\cdot n)/3}\cdot\left(\frac{1}{3}\right)!}{\left(\frac{1}{3} - n\right)!}$

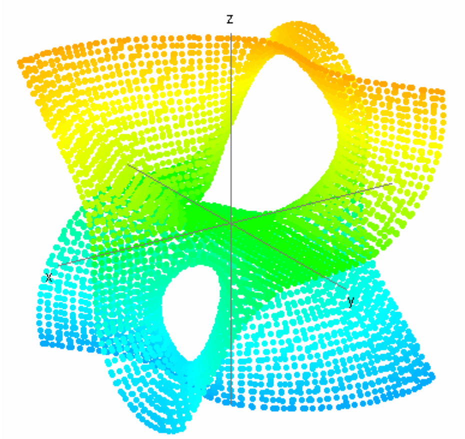
#7: $g\left(\frac{1}{3}, 10\right) = - \frac{2504902400}{59049\cdot(x + 1)^{29/3}}$

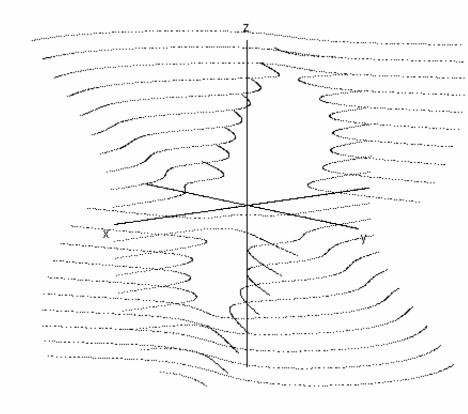
#8: $\frac{\left(\frac{1}{3}\right)!}{\left(\frac{1}{3} - 10\right)!}$

#9: $- \frac{2504902400}{59049}$



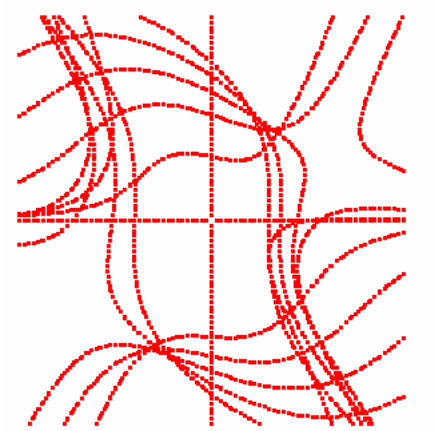
Surface #1: $x^3y + y^3z + z^3x + 7z^2 + 5z = 0$

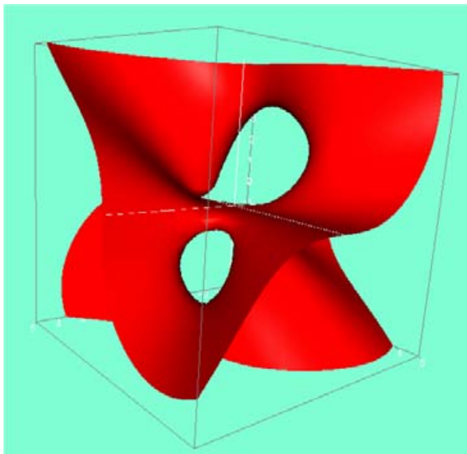




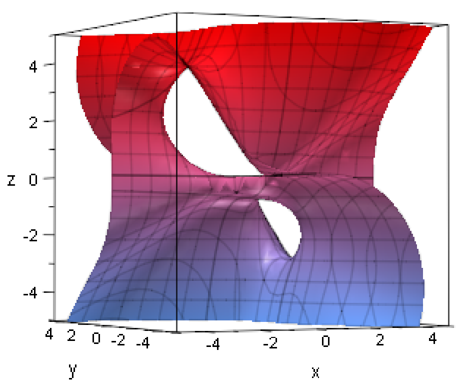
Surface #1 in DERIVE presented applying functions from `polycontour.mth` (DNL#63) and `IMPLICIT_Peter.mth` (DNL#64).

```
ImplicitPts(x^3·y + y^3·z + z^3·x + 7·z^2 + 5·z = 0, -4, 4, 0.2)

VECTOR(ContourPts_XY(x^3·y + y^3·z + z^3·x + 7·z^2 + 5·z = 0, z_, -4, 4, -4, 4, 0.2, 0.2), z_, -4, 4, 0.2)

VECTOR(ContourDots_XY(x^3·y + y^3·z + z^3·x + 7·z^2 + 5·z, 1), 1, -5, 5, 0.5)

VECTOR(ContourPts_2D(x^3·y + y^3·z + z^3·x + 7·z^2 + 5·z, 1), 1, -5, 5)
```

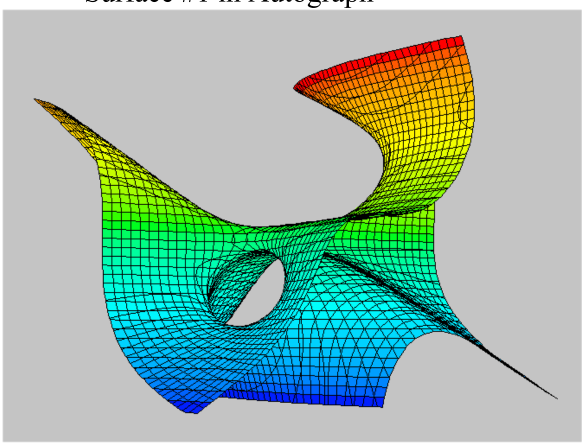




Surface #1 in Autograph



Surface #1 in MuPad



Surface #1 in DPGraph